

This Page Is Inserted by IFW Operations  
and is not a part of the Official Record

## **BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

## **IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning documents *will not* correct images,  
please do not report the images to the  
Image Problem Mailbox.**

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-139935

(43)Date of publication of application : 31.05.1996

(51)Int.Cl.

H04N 1/41

H03H 17/02

H03M 7/40

H04N 7/30

(21)Application number : 07-237987

(71)Applicant : RICOH CO LTD

(22)Date of filing : 18.09.1995

(72)Inventor : AAMADO ZANDEI  
JIEIMUSU DEII AREN  
EDOWAADO ERU SHIYUWARU  
MAATEIN PII BOORITSUKU

(30)Priority

Priority number : 94 310146 Priority date : 21.09.1994 Priority country : US

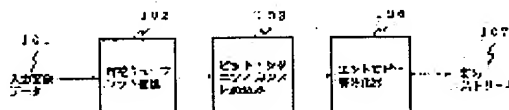
## (54) CODER AND WAVELET CONVERSION FILTER

(57)Abstract:

PURPOSE: To provide an efficient coder employing conversion to obtain excellent energy concentration and the wavelet conversion filter for the purpose.

CONSTITUTION: A reversible wavelet conversion block 102 divides image data 101 by using a non minimum length reversible filter to convert divisions into a code series. A bit significance imbedded block 103 converts the coefficient series into a form of code-absolute value and applies a frequency base model or an integrated space/frequency base model to the converted series.

The obtained decision is subjected to entropy coding by an entropy coder 104.



## LEGAL STATUS

[Date of request for examination] 10.04.2001

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

3436832

[Date of registration]

06.06.2003

[Number of appeal against examiner's decision  
of rejection]

[Date of requesting appeal against examiner's  
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-139935

(43) 公開日 平成8年(1996)5月31日

(51) Int.Cl. <sup>8</sup>	識別記号	庁内整理番号	F I	技術表示箇所
H 0 4 N 1/41	B			
H 0 3 H 17/02	B	8842-5J		
H 0 3 M 7/40		9382-5K		
H 0 4 N 7/30				
			H 0 4 N 7/ 133	Z
			審査請求 未請求	請求項の数24 O L (全 71 頁)

(21) 出願番号 特願平7-237987

(22) 出願日 平成7年(1995)9月18日

(31) 優先権主張番号 08/310146

(32) 優先日 1994年9月21日

(33) 優先権主張国 米国 (US)

(71) 出願人 000006747

株式会社リコー

東京都大田区中馬込1丁目3番6号

(72) 発明者 アーマド・ザンディ

アメリカ合衆国 カリフォルニア州

94025 メンローパーク サンド ヒル

ロード 2882 リコーコーポレーション内

(72) 発明者 ジェイムス ディー アレン

アメリカ合衆国 カリフォルニア州

94025 メンローパーク サンド ヒル

ロード 2882 リコーコーポレーション内

(74) 代理人 弁理士 鈴木 誠 (外1名)

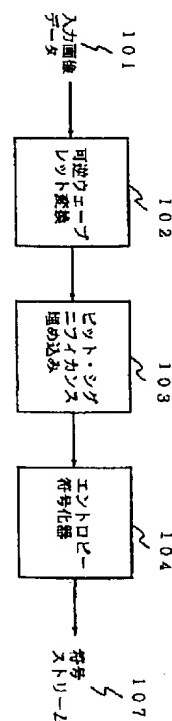
最終頁に続く

(54) 【発明の名称】 符号化装置及びウェーブレット変換フィルタ

(57) 【要約】

【課題】 良好なエネルギー集中を得られる変換を用いた効率的な符号化装置と、そのためのウェーブレット変換フィルタを提供する。

【解決手段】 可逆ウェーブレット変換ブロック102は画像データ101を非最小長可逆フィルタを用いて分割し符号系列に変換する。ビット・シグニフィカンス埋め込みブロック103は、その係数系列を符号-絶対値形式に変換した後に周波数ベースモデルまたは統合空間/周波数ベースモデルを適用する。得られたデシジョンをエントロピー符号化器104でエントロピー符号化する。



1

## 【特許請求の範囲】

【請求項 1】 入力データを圧縮データストリームに符号化するための符号化装置であって、  
該入力データを複数の係数に変換する可逆ウェーブレットフィルタ、

該複数の係数に対し埋め込み符号化を行なうことにより、ビットストリームを生成する、該可逆ウェーブレットフィルタに接続された埋め込み符号化器、及び該ビットストリームに対しエントロピー符号化を行なう、符号化データを生成する、該埋め込み符号化器に接続されたエントロピー符号化器からなる符号化装置。

【請求項 2】 入力データを符号化するための符号化装置であって、

該入力データを受け取って、該入力データの分割したものを表わす係数の系列を生成する変換符号化器、及び該係数の系列を受け取り、該係数の系列に対しビット・シグニフィカンス符号化を行なう符号化データを生成する埋め込み符号化器からなり、該埋め込み符号化器が該係数の系列の全部を受け取る前に符号化データを発生する符号化装置。

【請求項 3】 請求項 2 記載の符号化装置において、該変換符号化器及び該埋め込み符号化器は、1 パスで該入力データから符号化データを生成するように動作することを特徴とする符号化装置。

【請求項 4】 入力データ信号のフィルタリングのためのウェーブレット変換フィルタであって、

該入力データ信号のサンプルの第 1 のペアを加算して第 1 の結果を生成する第 1 加算器、

該第 1 の結果に基づいた第 1 の低域通過係数を出力する第 1 出力論理、

ある低域通過係数と該第 1 低域通過係数の入力を受け、該ある低域通過係数を該第 1 低域通過係数から減算して第 2 の結果を発生する第 1 減算器、ここにおいて該ある低域通過係数は該第 1 減算器にフィードバックとして受け取られる、

該入力データ信号のサンプルの第 2 のペアを相互に減算して第 3 の結果を発生する第 2 減算器、

該第 3 結果に基づく入力と該第 2 結果を受け取って加算し第 4 の結果を発生する第 2 加算器、

該第 4 結果に基づいた第 2 の低域通過係数を生成する第 2 出力論理からなり、該第 1 及び第 2 の低域通過係数を出力するウェーブレット変換フィルタ。

【請求項 5】 請求項 4 記載のウェーブレット変換フィルタにおいて、該第 1 出力論理は除算器からなることを特徴とするウェーブレット変換フィルタ。

【請求項 6】 請求項 5 記載のウェーブレット変換フィルタにおいて、該除算器はビットシフトからなることを特徴とするウェーブレット変換フィルタ。

【請求項 7】 請求項 4 記載のウェーブレット変換フィルタにおいて、該第 2 出力論理は除算器からなることを

2

特徴とするウェーブレット変換フィルタ。

【請求項 8】 請求項 7 記載のウェーブレット変換フィルタにおいて、該除算器はビットシフトからなることを特徴とするウェーブレット変換フィルタ。

【請求項 9】 入力データ信号のフィルタリングのためのウェーブレット変換フィルタであって、

該入力データ信号のサンプルの第 1 のペアを加算して第 1 の結果を生成する第 1 加算器、

該第 1 結果を受け取り、該第 1 結果に第 1 の因数を乗じて第 2 の結果を得る第 1 乗算器、ここにおいて該第 2 結果は第 1 の低域通過係数として出力される、

ある低域通過係数を受け取り該第 2 結果より減算して第 3 の結果を発生する第 1 減算器、ここにおいて、該ある低域通過係数は該第 1 減算器にフィードバックとして受け取られる、

該入力データ信号のサンプルの第 2 のペアを相互に減算して第 4 の結果を発生する第 2 減算器、

該第 4 結果を受け取って該第 4 結果を第 2 の因数で除して第 5 の結果を発生する第 1 除算器、

該第 3 結果及び該第 5 結果を受け取って加算し第 6 の結果を発生する第 2 加算器、及び該第 6 結果を受け取り該第 6 結果を第 3 の因数で除して第 2 の低域通過係数を生成する第 2 除算器からなり、該第 1 低域通過係数及び該第 2 低域通過係数が出力されるウェーブレット変換フィルタ。

【請求項 10】 請求項 9 記載のウェーブレット変換フィルタにおいて、該乗算器はシフトからなることを特徴とするウェーブレット変換フィルタ。

【請求項 11】 請求項 9 記載のウェーブレット変換フィルタにおいて、該除算器のすくなくとも一つはシフトからなることを特徴とするウェーブレット変換フィルタ。

【請求項 12】 入力データ信号のフィルタリングのためのウェーブレット変換フィルタであって、

1 対の低域通過係数の減算をして第 1 の結果を発生する第 1 減算器、

該第 1 結果に基づいた第 1 の入力を、ある高域通過係数から減算して第 2 の結果を発生する第 2 減算器、

該第 2 結果を、ある低域通過係数に基づいた第 2 入力に加算して第 3 の結果を発生する第 1 加算器、

該第 2 結果を、該ある低域通過係数から減算して第 4 の結果を発生する第 3 減算器からなり、該第 3 結果に基づいた第 1 のサンプルと該第 4 結果に基づいた第 2 のサンプルを出力するウェーブレット変換フィルタ。

【請求項 13】 該第 3 結果を受け取り、該第 3 結果をクリップして該第 1 サンプルを発生する第 1 クリップ機構をさらに具備することを特徴とする請求項 12 記載のウェーブレット変換フィルタ。

【請求項 14】 該第 1 結果を受け取り、該第 1 結果を第 1 因数により除して該第 1 入力を発生する除算器をさ

3

らに具備することを特徴とする請求項 12 記載のウェーブレット変換フィルタ。

【請求項 15】 該ある低域通過係数を受け取り、該ある低域通過係数を乗算して該第 2 入力を生成する乗算器をさらに具備することを特徴とする請求項 12 記載のウェーブレット変換フィルタ。

【請求項 16】 複数のデータシンボルを符号化するための符号化装置であって、  
該複数のデータシンボルを受け取って該複数のデータシンボルをフォーマットされたデータシンボルの集合にフ  
ォーマットするフォーマッティングユニット、  
該複数のデータシンボルに応じて複数のデシジョンを生成する、該フォーマッティングユニットに接続されたシ  
グニフィカンスユニット、及び該シグニフィカンスユ  
ニットから該複数のデシジョンを受け取って記憶または出力するメモリ機構からなる符号化装置。

【請求項 17】 請求項 16 記載の符号化装置において、該複数のデータシンボルは複数の係数からなることを特徴とする符号化装置。

【請求項 18】 請求項 16 記載の符号化装置において、該フォーマッティングユニットは、該複数のデータシンボルを符号-絶対値形式にフォーマットされたデータの集合へ変換するための符号-絶対値ユニットからなることを特徴とする符号化装置。

【請求項 19】 請求項 18 記載の符号化装置において、該符号-絶対値ユニットは、  
各データシンボルに対するシグニフィカンス表示を出力するシグニフィカンス判定機構、  
該複数のデータシンボルのそれぞれを受け取り、そのデータシンボルに対する符号及び仮数を生成する符号及び  
仮数ジェネレータ、及び該複数のデータシンボルのそれぞれに対するインデックスを生成するインデックスカウンタからなることを特徴とする符号化装置。

【請求項 20】 請求項 19 記載の符号化装置において、該シグニフィカンス判定機構はプライオリティエンコードからなることを特徴とする符号化装置。

【請求項 21】 請求項 19 記載の符号化装置において、該シグニフィカンス判定機構は、1 対のプライオリティエンコードと、該 1 対のプライオリティエンコードのそれぞれの出力の間で選択することによって該シグニフィカンス表示を出力する選択機構とからなることを特徴とする符号化装置。

【請求項 22】 該複数のフォーマットされたデータシンボルを受け取って記憶する絶対値メモリをさらに具備することを特徴とする請求項 16 記載の符号化装置。

【請求項 23】 請求項 22 記載の符号化装置において、該複数のフォーマットされたデータシンボルのそれぞれは、シグニフィカンスレベルに基づいて該絶対値メモリに格納されることを特徴とする符号化装置。

【請求項 24】 請求項 22 記載の符号化装置において

4

て、該絶対値メモリは複数の記憶領域を具備し、該複数の記憶領域のそれぞれは別個のシグニフィカンスレベルのためのデータを格納し、

該絶対値メモリは複数のカウンタをさらに具備し、該複数のカウンタのそれぞれは、該フォーマットされたデータシンボルの格納のためのアドレスを提供するため該複数の記憶領域の 1 つに関係付けられることを特徴とする符号化装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明はデータ圧縮及び伸長システムの分野に係り、特に、圧縮／伸長システムにおけるデータの非損失性 (lossless) 及び損失性 (lossy) 符号化及び復号のための方法及び装置に関する。

【0002】

【従来の技術】データ圧縮は、大量のデータの蓄積及び伝送のために非常に有用なツールである。例えば、文書のファクシミリ伝送のような画像の伝送に要する時間は、圧縮を使ってその画像の再生に必要とされるビット数を減らすと、飛躍的に短縮される。

【0003】従来より、多くの様々なデータ圧縮手法が存在している。圧縮手法は 2 つの大まかなカテゴリー、つまり損失性符号化と非損失性符号化とに分類できる。損失性符号化とは、情報の損失を生じ、したがってオリジナルデータの完全な再現が保証されない符号化のことである。損失性符号化の目標とするところは、オリジナルデータから変化しても、その変化が不快であったり目だったりしないようにすることである。非損失性圧縮では、全ての情報が保存され、データは完全な再現が可能な方法で圧縮される。

【0004】非損失性圧縮では、入力シンボルもしくは輝度データが出力符号語に変換される。入力としては、画像データ、音声データ、1 次元データ (例えば空間的または時間的に変化するデータ)、2 次元データ (例えば 2 つの空間軸方向に変化する (または 1 つの空間次元と 1 つの時間次元で変化する) データ)、あるいは多次元／マルチスペクトルのデータがあろう。圧縮がうまくいけば、その符号語は、符号化されない入力シンボル (または輝度データ) のために必要とされるビット数より少ないビット数で表現される。

【0005】非損失性符号化法には、辞書符号化方式 (例えば、Lempel-Ziv 方式)、ランレングス符号化方式、計数符号化方式、エントロピー符号化方式がある。非損失性画像圧縮では、圧縮は予測またはコンテキストと符号化に基づいている。ファクシミリ圧縮用の JBIG 規格と、連続濃淡画像用の DPCM (differential pulse code modulation-JPEG 規格のオプション) は画像用非損失性圧縮の例である。

【0006】損失性圧縮では、入力シンボルまたは輝度データは、量子化されてから出力符号語へ変換される。量

50

子化は、データの重要な特徴量を保存する一方、重要でない特徴量を除去することを目的としている。損失性圧縮システムは、量子化に先立ち、エネルギー集中をするために変換を用いる。

【0007】画像信号処理における最近の開発は、効率的かつ高精度のデータ圧縮符号化方式を追求することに関心を集中し続けている。変換またはピラミッド信号処理の様々な方式が提案されており、その中にマルチ解像度ピラミッド処理方式とウェーブレット (wavelet) ピラミッド処理方式とがある。これら2方式は、サブバンド処理方式及び階層処理方式とも呼ばれる。画像データのウェーブレット・ピラミッド処理方式は、直交ミラーフィルタ (QMF) を用いてオリジナル画像のサブバンド分割をする特殊なマルチ解像度ピラミッド処理方式である。なお、他の非QMFウェーブレット方式が存在する。

【0008】ウェーブレット処理方式に関し、より多くの情報を得るには、Antonini, M., et al., "Image Coding Using Wavelet Transform", IEEE Transactions on Image Processing, Vol.1, No.2, April 1992, 及び Shapiro, J., "An Embedded Hierarchical Image Coder Using Zerotrees of Wavelet Coefficients", Proc. IEEE Data Compression Conference, pgs.214-223, 1993 を参照されたい。

【0009】従来のウェーブレット処理方式の多くの問題点は、データ全部をその処理中に記憶しておくために大きなメモリが必要となることである。換言すれば、ウェーブレット処理を実施する場合、データ全部を、その符号化がなされる前に調べなければならない。かかる場合、少なくとも1回、データ全部のフルパスを完了するまでは全くデータ出力がない。実際、ウェーブレット処理は普通、データのマルチパスを必要とする。それ故に、大きなメモリがしばしば必要になる。大きなメモリを必要とせずに、ウェーブレット処理を利用することが望ましい。さらに、データの単一パスだけでウェーブレット処理を遂行することが望ましい。

【0010】多くのウェーブレットまたはサブバンド変換装置は特定の正規型のフィルタを必要とする。例えば、低域通過フィルタと高域通過フィルタは同一長でなければならない、その係数の2乗和は1でなければならない、高域通過フィルタは時間一周波数特性が低域通過フィルタと正反対でなければならない、等々である。(1991年5月にLawton等に発行された米国特許第5,014,134号を参照)。より広範な種類のフィルタを許容することが望まれる。すなわち、低域通過フィルタと高域通過フィルタは同一長でなく、その係数の2乗和は1である必要がなく、高域通過フィルタは低域通過フィルタと正反対の時間一周波数特性である必要がない等々、そのような低域通過フィルタと高域通過フィルタを使用するウェーブレットまたはサブバンド変換装置を提

供することが望まれる。

#### 【0011】

【発明が解決しようとする課題】本発明は、前述の問題点を鑑み、効率的な非圧縮性／圧縮性データ圧縮／伸長システムを実現しようとするもので、より具体的には、良好なエネルギー集中を得られる変換を用いた効率的な符号化装置と、それに関連した効率的な圧縮を可能にするウェーブレット変換フィルタとを提供することを目的とする。

#### 【0012】

【課題を解決するための手段】本発明によれば、入力データを複数の係数に変換する可逆ウェーブレットフィルタと、該複数の係数に対し埋め込み符号化を行なうことにより、ビットストリームを生成する、該可逆ウェーブレットフィルタに接続された埋め込み符号化器と、該ビットストリームに対しエントロピー符号化を行なって、符号化データを生成する、該埋め込み符号化器に接続されたエントロピー符号化器からなる符号化装置が提供される。

【0013】本発明によれば、入力データを受け取って、該入力データの分割したものを表わす係数の系列を生成する変換符号化器と、該係数の系列を受け取り、該係数の系列に対しビット・シグニフィカンス符号化を行なって符号化データを生成する埋め込み符号化器とからなり、該埋め込み符号化器が該係数の系列の全部を受け取る前に符号化データを発生する符号化装置が提供される。この符号化装置は例えば、該変換符号化器及び該埋め込み符号化器が1パスで該入力データから符号化データを生成するように動作するように構成される。

【0014】本発明によれば、入力データ信号のサンプルの第1のペアを加算して第1の結果を生成する第1加算器と、該第1の結果に基づいた第1の低域通過係数を出力する第1出力論理と、ある低域通過係数と該第1低域通過係数の入力とを有し、該ある低域通過係数を該第1低域通過係数から減算して第2の結果を発生する第1減算器 (該ある低域通過係数は該第1減算器にフィードバックとして受け取られる) と、該入力データ信号のサンプルの第2のペアを相互に減算して第3の結果を発生する第2減算器と、該第3結果に基づく入力と該第2結果を受け取って加算し第4の結果を発生する第2加算器と、該第4結果に基づいた第2の低域通過係数を生成する第2出力論理とからなり、該第1及び第2の低域通過係数を出力するウェーブレット変換フィルタが提供される。このウェーブレット変換フィルタにおいて、該第1出力論理及び／または該第2出力論理は、例えば除算器からなり、該除算器は例えばビットシフトからなる。

【0015】本発明によれば、入力データ信号のサンプルの第1のペアを加算して第1の結果を生成する第1加算器と、該第1結果を受け取り、該第1結果に第1の因数を乗じて第2の結果を得る第1乗算器 (該第2結果は

7

第1の低域通過係数として出力される)と、ある低域通過係数を受け取り該第2結果より減算して第3の結果を発生する第1減算器(該ある低域通過係数は該第1減算器にフィードバックとして受け取られる)と、該入力データ信号のサンプルの第2のペアを相互に減算して第4の結果を発生する第2減算器と、該第4結果を受け取って該第4結果を第2の因数で除して第5の結果を発生する第1除算器と、該第3結果及び該第5結果を受け取って加算し第6の結果を発生する第2加算器と、該第6結果を受け取り該第6結果を第3の因数で除して第2の低域通過係数を生成する第2除算器とからなり、該第1低域通過係数及び該第2低域通過係数が出力されるウェーブレット変換フィルタが提供される。このウェーブレット変換フィルタにおいて、該乗算器及び/または該除算器の少なくとも1つは例えばシフタからなる。

【0016】本発明によれば、1対の低域通過係数の減算をして第1の結果を発生する第1減算器と、該第1結果に基づいた第1の入力を、ある高域通過係数から減算して第2の結果を発生する第2減算器と、該第2結果を、ある低域通過係数に基づいた第2入力に加算して第3の結果を発生する第1加算器と、該第2結果を、該ある低域通過係数から減算して第4の結果を発生する第3減算器とからなり、該第3結果に基づいた第1のサンプルと該第4結果に基づいた第2のサンプルを出力するウェーブレット変換フィルタが提供される。このウェーブレット変換フィルタに、該第3結果を受け取り、該第3結果をクリップして該第1サンプルを発生する第1クリップ機構、及び/または、該第1結果を受け取り、該第1結果を第1因数により除して該第1入力を発生する除算器、及び/または、該ある低域通過係数を受け取り、該ある低域通過係数を乗算して該第2入力を生成する乗算器を加えた構成のウェーブレット変換フィルタも本発明により提供される。

【0017】本発明によれば、複数のデータシンボルを受け取って該複数のデータシンボルをフォーマットされたデータシンボルの集合にフォーマットするフォーマティングユニットと、該複数のデータシンボルに応じて複数のデジションを生成する、該フォーマティングユニットに接続されたシグニフィカンスユニットと、該シグニフィカンスユニットから該複数のデジションを受け取って記憶または出力するメモリ機構とからなる符号化装置が提供される。この符号化装置において、該複数のデータシンボルは例えば複数の係数からなり、該フォーマティングユニットは、例えば、該複数のデータシンボルを符号-絶対値形式にフォーマットされたデータの集合へ変換するための符号-絶対値ユニットからなり、該符号-絶対値ユニットは、例えば、各データシンボルに対するシグニフィカンス表示を出力するシグニフィカンス判定機構と、該複数のデータシンボルのそれぞれを受け取り、そのデータシンボルに対する符号及び仮数を

8

生成する符号及び仮数ジェネレータと、該複数のデータシンボルのそれぞれに対するインデックスを生成するインデックスカウンタとからなる。該シグニフィカンス判定機構は、例えば、プライオリティエンコードからなり、あるいは、1対のプライオリティエンコードと、該1対のプライオリティエンコードのそれぞれの出力の間で選択することによって該シグニフィカンス表示を出力する選択機構とからなる。

【0018】また、上記構成の符号化装置に、該複数のフォーマットされたデータシンボルを受け取って記憶する絶対値メモリを追加した構成の符号化装置も本発明により提供される。この符号化装置において、例えば、該複数のフォーマットされたデータシンボルのそれぞれは、シグニフィカンスレベルに基づいて該絶対値メモリに格納される。また、例えば、該絶対値メモリは複数の記憶領域を具備し、該複数の記憶領域のそれぞれは別個のシグニフィカンスレベルのためのデータを格納し、該絶対値メモリは複数のカウンタをさらに具備し、該複数のカウンタのそれぞれは、該フォーマットされたデータシンボルの格納のためのアドレスを提供するため該複数の記憶領域の1つに関係付けられる。

【0019】

【発明の実施の態様】本発明は、入力データに応じて変換信号を生成する方法及び装置を含む。一実施例において、この変換信号は、可逆ウェーブレット変換によって生成される。本発明はまた、変換信号を、入力データの非損失圧縮されたものを表わすデータへと圧縮するための方法及び装置を含む。一実施例において、本発明は非最小長の可逆フィルタを用いて入力データを分割する。この分割は、多数の1次元フィルタを用いて行なわれるかもしれない。

【0020】本発明はまた、変換信号の埋め込み符号化(embedded coding)を行なう方法及び装置を含む。本発明の埋め込み符号化は、係数系列の順序付けと変換信号に対するビット・シグニフィカンス埋め込み(bit significance embedding)を含む。

【0021】本発明はまた、入力データの非損失圧縮データを変換信号へ伸長するための方法及び装置も含む。本発明はまた、非損失性圧縮データの丸めにより入力信号の損失性圧縮も可能である。本発明は、インバース可逆ウェーブレット変換を用いて、変換信号から入力データを生成し、入力データの再現データを得るための方法及び装置も含む。

【0022】

【実施例】圧縮及び伸長のための方法及び装置について述べる。以下の本発明に関する詳細な説明において、本発明を完全に理解してもらうために、符号化器の種類、ビット数、信号名等々、様々な具体例が示される。しかし、当業者にとって、そのような具体例によらずに本発明を実施し得ることは明白であろう。他方、本発明をい



たづらに難解にしないため、周知の構造及びデバイスは詳細にはなくブロック図の形で示される。

【0023】以下の詳細説明のかなりの部分が、コンピュータメモリ内のデータビットに対する演算のアルゴリズム及び記号表現によって与えられる。これらのアルゴリズム記述及び表現は、データ処理技術分野の当業者によって、その研究の内容を他の当業者に対し最も効率的に伝えるために用いられる手段である。あるアルゴリズムがあり、それが概して、希望する結果に至る自己矛盾のないステップ系列だと考えられるとしよう。これらのステップは、物理量の物理的处理を必要とするものである。必ずという訳ではないが、通常、これらの物理量は記憶、転送、結合、比較、その他処理が可能な電氣的または磁氣的信号の形をとる。これらの信号をビット、値、要素、記号、文字、用語、数字等で表わすのが、主に慣用上の理由から、時に都合がよいことが分かっている。

【0024】しかしながら、このような用語は、適切な物理量と関係付けられるべきであって、これら物理量につけた便宜上のラベルに過ぎないということを心に留めるべきである。以下の説明から明らかなように、特に断わらない限り、“処理” “演算” “計算” “判定” “表示” 等々の用語を用いて論じることが、コンピュータシステムのレジスタ及びメモリ内の物理的（電子的）な量として表現されたデータを処理して、コンピュータシステムのメモリまたはレジスタ、同様の情報記憶装置、情報伝送装置あるいは表示装置の内部の同様に物理量として表現された他のデータへ変換する、コンピュータシステムあるいは同様の電子演算装置の作用及びプロセスを指すものである。

【0025】本発明はまた、本明細書に述べられる操作を実行する装置に関する。この装置は、必要な目的のために専用に作られてもよいし、あるいは、汎用コンピュータを内蔵プログラムにより選択的に駆動または再構成したものでもよい。本明細書に提示されるアルゴリズム及び表示は、本質的に、いかなる特定のコンピュータやその他装置とも関係がない。様々な汎用マシンを本明細書に教示したところに従ったプログラムと一緒に用いてもよいし、あるいは、必要な方法のステップの実行のためにより特化した装置を作るほうが好都合であるかもしれない。これら多様なマシンに要求される構造は以下の説明より明らかになろう。さらに、本発明は、特定のプログラミング言語と関連付けでは説明されない。本明細書において述べるように、本発明の教示するところを実現するために、多様なプログラミング言語を用い得ることが理解されるであろう。

【0026】＜概要＞本発明は、符号化部及び復号部を持つ圧縮／伸長システムを提供する。符号化部は入力データを符号化して圧縮データを生成する役割を持ち、他方、復号部は前もって符号化されたデータを復号してオ

リジナル入力データの再現データを生成する役割を持つ。入力データは、画像（静止画像あるいはビデオ画像）、音声等々の様々なデータ形式のものがあろう。一実施例ではデータはデジタル信号データであるが、デジタル化されたアナログデータ、テキストデータ形式、その他の形式が可能である。データのソースは、符号化部及び／または復号部のためのメモリまたはチャンネルであらう。

【0027】本発明における符号化部及び／または復号部の構成要素は、ハードウェアあるいは、コンピュータシステム上で使われるようなソフトウェアによって実現し得る。本発明は、非損失性圧縮／伸長システムを提供する。本発明はまた、損失性圧縮／伸長を実行するようにも構成し得る。

【0028】図1は、システムの符号化部の一実施例のブロック図である。なお、システムの復号部はデータフローに沿って逆の順序で動作する。

【0029】図1において、入力画像データ101が可逆ウェーブレット変換ブロック102に受け取られる。可逆ウェーブレット変換ブロック102の出力はビット・シグニフィカンス埋め込み (bit-significance embedding) ブロック103に接続される。可逆ウェーブレット変換ブロック102の出力にตอบสนองして、ビット・シグニフィカンス埋め込みブロック103は少なくとも1つのビットストリームを出力し、このビットストリームはエントロピー符号化器104に受け取られる。エントロピー符号化器104は、ビット・シグニフィカンス埋め込みブロック103からの入力に応じて、符号ストリーム107を出力する。

【0030】一実施例において、ビット・シグニフィカンス埋め込みブロック103は、図2に示されるように、符号 (sign) 一絶対値フォーマットユニット109、周波数ベース (frequency-based) コンテキストモデル105及び統合空間／周波数 (joint space/frequency) コンテキストモデル106からなる。一実施例では、統合空間／周波数コンテキストモデル106は水平 (horizon) コンテキストモデルである。いくつかの実施例では、周波数ベースコンテキストモデル105はシグニフィカンス・ツリー (significance tree) モデルである。符号一絶対値フォーマットユニット109、周波数ベースコンテキストモデル105及び統合空間／周波数 (JSF) コンテキストモデル106は、本発明におけるビット・シグニフィカンス埋め込みを遂行する。符号一絶対値フォーマットユニット109の入力は可逆ウェーブレット変換ブロック102の出力と接続される。符号一絶対値フォーマットユニット109の出力はスイッチ108に接続される。スイッチ108は、符号一絶対値フォーマットユニット109の出力を周波数ベースコンテキストモデル (モデリングブロック) 105または統合空間／周波数

11

コンテキストモデル（統合空間／周波数モデリングブロック）106のいずれかの入力に与えるように接続される。周波数ベースコンテキストモデル（周波数ベース符号化ブロック）105及び統合空間／周波数コンテキストモデル（水平順序（horizon order）符号化ブロック）106の出力はエントロピー符号化器104の入力に接続される。エントロピー符号化器104は、出力符号ストリーム107を生成する。

【0031】図1に戻る。本発明においては、後に定義するように、画像データ1014は可逆ウェーブレット変換ブロック102において受け取られ、可逆ウェーブレットを使って変換符号化され、その画像のマルチ解像度分割を表わす一つの係数系列が生成される。これらの係数はビット・シグニフィカンス埋め込みブロック103に受け取られる。

【0032】ビット・シグニフィカンス埋め込みブロック103は、それらの係数を順序付けして符号-絶対値形式へ変換する。そして、このフォーマットされた係数は、それらのシグニフィカンス（significance）（後述）に基づき、様々な埋め込み（embedded）モデリング法の組合せを適用される。本発明においては、フォーマットされた係数は2つの埋め込みモデリング法（例えば周波数ベースモデリングとJSFモデリング）の中のいずれか一方を適用される。

【0033】一実施例においては、フォーマットされた係数は、周波数ベースモデリングか統合空間／周波数モデリングを適用される。本発明において、入力データが複数のビットプレーン（bitplane）を持つ画像データからなる時には、いくつかのビットプレーンが周波数モデリングによって符号化され、残りのビットプレーンがJSFモデリングにより符号化される。どのビットプレーンにどの方法を使うべきかの決定は、ユーザ・パラメータであってよい。一実施例においては、上位の係数ビットプレーンが順序付けられて本発明の周波数ベースモデリングにより符号化される。本発明の周波数ベースコンテキストモデル法においては、係数ビットのシグニフィカンスの予測はウェーブレットのピラミッド構造と関連している。下位の係数ビットプレーンは順序付けられて本発明の統合空間／周波数コンテキストモデルにより符号化される。このJSFモデリング（例えば水平モデリング）は、周波数ベース符号化に比べ、周波数領域係数関係の相関が小さいビットプレーンに対して有利である。

【0034】ビット・シグニフィカンス埋め込みの結果が、エントロピー符号化器により符号化すべきデシジョン（decision）（あるいはシンボル）である。一実施例では、全てのデシジョンが一つの符号化器へ送られる。他の実施例では、デシジョンはシグニフィカンス（significance）によってラベル付けされ、各シグニフィカンス・レベルのデシジョンが別々の複数の（物理または仮

12

想）符号化器によって処理される。

【0035】周波数ベースコンテキストモデル105及びJSFモデル106より得られたビットストリームは、シグニフィカンス順に、エントロピー符号化器104によって符号化される。一実施例では、エントロピー符号化器104はバイナリのエントロピー符号化器からなる。ある実施例では、エントロピー符号化器104はQコード、米国特許第5,272,478号に定義されたBコード、あるいは、米国特許出願第08/016,035号（"Method and Apparatus for Parallel Decoding and Encoding of Data"、1993年2月10日受理）に述べられているような符号化器からなる。Qコードについてさらに情報を得るには、Pennebaker, W. B., et al., "An Overview of the Basic Principles of the Q-coder Adaptive Binary Arithmetic, IBM Journal of Research and Development Vol. 32, pg. 717-26, 1988 を参照されたい。ある実施例では、単一の符号化器が単一の出力符号ストリームを発生する。他の実施例では、複数の（物理または仮想）符号化器が、複数の（物理または仮想）データストリームを発生する。

【0036】＜ウェーブレット分割（Wavelet Decomposition）＞本発明は、最初に、可逆ウェーブレットを用いて、画像（画像データとしての）または他のデータ信号の分割を行なう。本発明においては、可逆ウェーブレット変換は、整数係数を持つ信号の非損失性復元が可能な精密復元システムを整数演算で実現する。本発明は、可逆ウェーブレットを用いることにより、限定された精度の演算によって非損失性圧縮を提供できる。画像データに可逆ウェーブレット変換を適用することによって生成される結果は、一つの係数系列である。本発明の一実施例では、可逆ウェーブレット変換はフィルタの集合を用いて実現される。ある実施例では、そのフィルタとは1つの2タップ低域通過フィルタと1つの6タップ高域通過フィルタである。ある実施例では、これらフィルタは加減算（及びハードワイヤのビットシフト）だけで実現される。また、本発明においては、高域通過フィルタは低域通過フィルタの結果を使って出力を生成する。結果として得られる高域通過係数は画素解像度より数ビット分だけ大きく、低域通過係数は画素解像度と同一である。ピラミッド分割では低域通過係数だけが繰り返しフィルタされるため、マルチレベル分割により解像度は増加しない。

【0037】ウェーブレット変換システムは一对のFIR分析フィルタ $h_0(n)$ ,  $h_1(n)$ と一对のFIR合成フィルタ $g_0(n)$ ,  $g_1(n)$ によって定義される。本発明において、 $h_0$ と $g_0$ は低域通過フィルタであり、 $h_1$ と $g_1$ は高域通過フィルタである。このウェーブレット変換システムのブロック図が、図3に示されている。

【0038】図3において、ブロック201, 202により入力信号 $x(n)$ に対して分析フィルタ $h_0$ ,  $h_1$ が

13

けられ、その出力がブロック203、204で2:1の間引き(臨界的サブサンプリング)を施されることにより、変換信号 $y_0(n)$ 、 $y_1(n)$ が生成される。この変換信号 $y_0(n)$ 、 $y_1(n)$ はそれぞれ、本明細書においては低域通過係数、高域通過係数と呼ばれる。これらの分析フィルタとそれらに対応した間引きもしくはサブサンプリングのブロック201~204は、ウェーブレット変換システムの分析部を構成する。符号化器/復号器205、206は、変換ドメインにおいて実行される全ての処理論理及びルーチン(例えば、予測、量子化、符号化等々)を含む。

【0039】図3に示したウェーブレット変換システムは合成部も持ち、この合成部において変換信号はブロック207、208により1:2のアップサンプリングを施され(各項の後にゼロが挿入される)、ついで合成フィルタ $g_0(n)$ 、 $g_1(n)$ (ブロック209、210)に通される。低域通過係数 $y_0(n)$ は低域通過合成フィルタ $g_0$ (ブロック209)に通され、高域通過係数 $y_1(n)$ は高域通過合成フィルタ $g_1$ (ブロック210)に通される。フィルタ $g_0(n)$ 、 $g_1(n)$ の出力が合成されて

【0040】

【外1】

$$\hat{x}(n)$$

【0041】が作られる。

$$\hat{X}(Z) = \frac{1}{2}[H_0(Z)G_0(Z)+H_1(Z)G_1(Z)]X(Z) + \frac{1}{2}[H_0(-Z)G_0(Z)+H_1(-Z)G_1(Z)]X(-Z)$$

【0049】本発明において、上記式における右辺の第2項は"エイリアシング"(折り返し)項と呼ばれるが、これはキャンセルされる。というのは、合成フィルタが分析フィルタの直交ミラーフィルタとされているからである。すなわち、

【0050】

【数2】

$$\begin{cases} G_0(Z)=H_1(-Z) \\ G_1(Z)=-H_0(-Z) \end{cases}$$

【0051】フィルタ係数によれば、

【0052】

【数3】

$$\begin{cases} g_0(n)=(-1)^n h_1(n) \\ g_1(n)=-(-1)^n h_0(n) \end{cases}$$

【0053】したがって、直交ミラーフィルタのペアに関し、代入すると、出力は次のようになる。

【0054】

【数4】

14

【0042】ある実施例ではダウンサンプリング(サブサンプリング)とアップサンプリングが行なわれるが、他の実施例では、ダウンサンプリング及びアップサンプリングにより不要となる計算が行なわれないようなフィルタが使用される。

【0043】ウェーブレット変換システムはZ変換によって記述してもよい。ここにおいて、

【0044】

【外2】

$$X(Z), \hat{X}(Z)$$

【0045】は入力信号と出力信号であり、 $Y_0$

$(Z)$ 、 $Y_1(Z)$ は低域通過の変換信号と高域通過の変換信号であり、 $H_0(Z)$ 、 $H_1(Z)$ は低域通過分析フィルタと高域通過分析フィルタであり、 $G_0(Z)$ 、 $G_1(Z)$ は低域通過合成フィルタと高域通過合成フィルタである。変換ドメインにおいて修正も量子化もなければ、図3の出力

【0046】

【外3】

$$\hat{X}(Z)$$

【0047】は次式により与えられる。

【0048】

【数1】

$$\hat{X}(z) = \frac{1}{2}[H_0(Z)H_1(-Z)-H_1(Z)H_0(-Z)]X(Z)$$

【0055】よって、本発明の直交ミラーシステムにおいては、出力は分析フィルタの項だけで決定される。ウェーブレット変換は、フィルタにより生成された出力がフィルタの入力として直接的または間接的に用いられるという点で、変換信号に対し再帰的に適用される。以上説明した実施例においては、低域通過変換成分 $y_0$

$(n)$ だけが再帰的に変換されるため、当該システムはピラミッド型である。そのようなピラミッドシステムの一例が図9に示されている。

【0056】Z変換はハードウェア及び/またはソフトウェアのデータ操作を説明するのに便利な表現方法である。Z<sup>-m</sup>による乗算は、ハードウェアによるmクロックサイクルの遅延、及び、ソフトウェアによるm個前の要素への配列アクセスのモデルである。そのようなハードウェア手段は、メモリ、パイプステージ、シフタ、レジスタ等を含む。

【0057】本発明において、信号 $x(n)$ と信号

15

【0058】

【外4】

$$\hat{x}(n)$$

【0059】は、ある乗定数及びある遅延期間までは一致する。すなわち、Z変換では

【0060】

【数5】

$$\hat{X}(Z) = cZ^{-m}X(Z)$$

【0061】これは精密復元システムと呼ばれる。しかし、本発明の一実施例において、入力データに最初に適用されたウェーブレット変換は精密に復元可能である。

【0062】ハダマード変換 (Hadamard Transform) を使う本発明の一実施例は精密復元システムであり、その正規化形式はZ領域で次のように記述される。

【0063】

【数6】

$$\begin{cases} H_0(Z) = \frac{1}{\sqrt{2}}(1+Z^{-1}) \\ H_1(Z) = \frac{1}{\sqrt{2}}(1-Z^{-1}) \end{cases}$$

【0064】代入すると、出力は次のとおりである。

【0065】

【数7】

$$\hat{X}(Z) = Z^{-1}X(Z)$$

【0066】これが精密復元であることは明らかである。ハダマード変換について更に情報を得るためには、Anil K. Jain, Fundamentals of Image Processing, p. 155を見られたい。

【0067】ハダマード変換の可逆型は、本明細書においてS-変換と呼ばれる。S-変換についてさらに情報を得るためには、Said, A. and Pearlman, W., "Reversible Image Compression via Multiresolution Representation and Predictive Coding", Dept. of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 1993 を参照されたい。ハダマード変換は精密復元変換であるから、下に示す非正規化型 (一定の因子だけがハダマード変換と違っている) もまた精密復元変換である。

【0068】

【数8】

$$\begin{cases} h_0(Z) = \frac{1}{2}(1+Z^{-1}) \\ h_1(Z) = 1-Z^{-1} \end{cases}$$

【0069】入力信号のサンプルを  $x_0$ ,  $x_1$  とすると、

16

S-変換はつぎのように当該システムの可逆システムである。

【0070】

【数9】

$$\begin{cases} y_0(0) = [(x(0)+x(1))/2] \\ y_1(0) = x(0) - x(1) \end{cases}$$

【0071】上記式中の記号

【0072】

【外5】

[.]

【0073】は、切り捨てる、あるいは丸めることを意味し、時にフロア関数 (floor function) と呼ばれる。同様に、シーリング関数 (ceiling function)

【0074】

【外6】

[.]

【0075】は最も近い整数へ切り上げることを意味する。

【0076】このシステムが可逆であることの証明は、近似により失われる情報が  $x(0) + x(1)$  の最下位ビットだけであるという事実から得られる。しかしながら、 $x(0) + x(1)$  と  $x(0) - x(1)$  の最下位ビットは同一であるので、これは高域通過出力  $y_1(0)$  から再生することができる。換言すれば、次式のとおりである。

【0077】

【数10】

$$\begin{cases} x(0) = y_0(0) + [(y_1(0)+1)/2] \\ x(1) = y_0(0) - [(y_1(0)-1)/2] \end{cases}$$

【0078】S-変換は、最小長 (minimal length) の可逆フィルタを用いる非オーバーラップ (non-overlapping) 変換である。最小長フィルタは、2タップのフィルタのペアからなる。最小長変換は良好なエネルギー集中を得られない。最小長フィルタは、その長さがフィルタの数に等しいので、非オーバーラップ変換を実現する。オーバーラップ変換は、フィルタ数より大きな長さのフィルタを少なくとも1つ用いる。長い (非最小長の) フィルタを使うオーバーラップ変換は、より良好なエネルギー集中を得ることができる。本発明は、オーバーラップ変換を可能にする非最小長の可逆フィルタを提供する。

【0079】精密復元システムのもう一つの例は、次に示すZ領域定義を持つTwo/Six (TS) -変換からなる。

【0080】

【数11】

17

$$\begin{cases} H_0(Z) = \frac{1}{\sqrt{2}}(1+Z^{-1}) \\ H_1(Z) = \frac{1}{8\sqrt{2}}(-1-Z^{-1}+8Z^{-2}-8Z^{-3}+Z^{-4}+Z^{-5}) \end{cases}$$

【0081】代入すると、出力は次の通りであり、これは精密復元変換である。

【0082】

【数12】

$$\hat{X}(Z) = 2Z^{-3}X(Z)$$

【0083】TS-変換の有理非正規化型は次のとおりである。

$$\begin{cases} y_0(0) = \lfloor (x(0)+x(1))/2 \rfloor \\ y_0(1) = \lfloor (x(2)+x(3))/2 \rfloor \\ y_0(2) = \lfloor (x(4)+x(5))/2 \rfloor \end{cases}$$

$$y_1(0) = \lfloor (-x(0)+x(1)) + 8(x(2)-x(3)) + (x(4)+x(5)) / 8 \rfloor$$

【0087】しかし、TS-変換の有理非正規化型をそのまま具体化したのでは可逆でない。次に述べる例で、そのような構成が局所的に非可逆であることを明らかにする。グローバルケースのための例としては、もっと長い系列を作ることができる。 $y_0(0)$ と $y_0(2)$ を計算するために丸めを行なうため、  
 $-(x(0)+x(1))+ (x(4)+x(5)) \neq -y_0(0) + y_0(2)$

$$y_0(0) = \lfloor (1+1)/2 \rfloor = 1$$

$$y_0(1) = \lfloor (3+1)/2 \rfloor = 2$$

$$y_0(2) = \lfloor (1+1)/2 \rfloor = 1$$

$$y_1(0) = \lfloor [-(1+1)+8(3-1)+(1+1)]/8 \rfloor = \lfloor (-2+16+2)/8 \rfloor = 2$$

【0090】また、 $x(0)=1$ ,  $x(1)=2$ ,  $x(2)=4$ ,  $x(3)=1$ ,  $x(4)=1$ ,  $x(5)=1$ であるときには、

$$y_0(0) = \lfloor (1+2)/2 \rfloor = 1$$

$$y_0(1) = \lfloor (4+1)/2 \rfloor = 2$$

$$y_0(2) = \lfloor (1+1)/2 \rfloor = 1$$

$$y_1(0) = \lfloor [-(1+2)+8(4-1)+(1+1)]/8 \rfloor = \lfloor (-3+24+2)/8 \rfloor = \lfloor 23/8 \rfloor = 2$$

【0092】 $y_0(0)$ ,  $y_0(1)$ ,  $y_1(0)$ は入力 $x(0)\dots x(5)$ の異なった2つの組に対して同一であるから、この変換は可逆でない。 $y_0(0), \dots, y_1(0)$ を与えられたとき、このローカル情報から、その2つの組のいずれが入力されたのか判定できないからである。(全係数からのグローバル情報を用いるときに、当該変換が可逆でないとは証明できないことに注意されたい。)

さて、異なった高域通過フィルタ作用を提供する可逆T

18

【0084】

【数13】

$$\begin{cases} h_0(Z) = \frac{1}{2}(1+Z^{-1}) \\ h_1(Z) = \frac{1}{8}(-1-Z^{-1}+8Z^{-2}-8Z^{-3}+Z^{-4}+Z^{-5}) \end{cases}$$

【0085】 $\hat{x}(0)$ ,  $\hat{x}(1)$ ,  $\dots$ ,  $\hat{x}(5)$ が信号の6サンプルであるとき、最初の3つの低域通過係数 $y_0(0)$ ,  $y_0(1)$ ,  $y_0(2)$ と最初の高域通過係数 $y_1(0)$ は次式で示される。

【0086】

【数14】

であるから、ローカル情報を使うとき当該変換は可逆でない。

【0088】例えば、 $x(0)=1$ ,  $x(1)=1$ ,  $x(2)=3$ ,  $x(3)=1$ ,  $x(4)=1$ ,  $x(5)=1$ であるときには、

【0089】

【数15】

【0091】

【数16】

S-変換を考えよう。この変換を、ここではRTS-変換と呼ぶ。

【0093】 $x(0)$ ,  $x(1)$ ,  $x(2)$ ,  $x(3)$ ,  $x(4)$ ,  $x(5)$ が信号の6サンプルであるとき、初めの3つの低域通過係数 $y_0(0)$ ,  $y_0(1)$ ,  $y_0(2)$ と最初の高域通過係数 $y_1(0)$ は次式で与えられる。

【0094】

【数17】

$$\begin{cases} y_0(0) = \lfloor (x(0)+x(1))/2 \rfloor \\ y_0(1) = \lfloor (x(2)+x(3))/2 \rfloor \\ y_0(2) = \lfloor (x(4)+x(5))/2 \rfloor \end{cases}$$

$$y_1(0) = \lfloor \lfloor (x(0)+x(1))/2 \rfloor + 4(x(2)-x(3)) + \lfloor (x(4)+x(5))/2 \rfloor \rfloor / 4$$

$$= \lfloor (-y_0(0) + 4(x(2)-x(3)) + y_0(2)) / 4 \rfloor$$

【0095】

【数18】

$$x(2)-x(3)=y_1(0)-\lfloor (-y_0(0)+y_0(2))/4 \rfloor$$

【0096】上式が成り立つので、 $x(2)-x(3)$ は完全に既知となった。前記(数17)の $y_0(1)$ と、 $x(2)-x(3)$ と $x(2)-x(3)$ とが上のように決まれば、 $x(0)+x(1)$ と $x(0)-x(1)$ の最下位ビットは同一であるから、 $x(2)$ と $x(3)$ は復元できる。すなわち、次式のとおりである。

【0097】

【数19】

$$d(0)=x(2)-x(3)=y_1(0)-\lfloor (-y_0(0)+y_0(2))/4 \rfloor$$

$$x(2)=y_0(1)+\lfloor (d(0)+1)/2 \rfloor$$

$$x(3)=y_0(1)+\lfloor (d(0)-1)/2 \rfloor$$

【0098】プログラミング言語“C”で実現した、RTS-変換のためのフィルタの例を図57乃至図60に示す。図57及び図58に示すものは1レベル分割RTS変換のためのフォワードフィルタ及びインバースフィルタであり、図59及び図60に示すものは2レベル分割RTS変換のためのフォワードフィルタ及びインバースフィルタである。

【0099】なお、数学的には

$$1/8 \cdot (-1 - Z^{-1} + 8Z^{-2} - 8Z^{-3} + Z^{-4} + Z^{-5})$$

と、

$$\begin{cases} h_0(Z) = \frac{1}{2}(1+Z^{-1}) \\ h_1(Z) = \frac{1}{4}\left(-\frac{1}{2}(1+Z^{-1}) + 4(Z^{-2}-Z^{-3}) + \frac{1}{2}(Z^{-4}+Z^{-5})\right) \end{cases}$$

【0103】このように、低域通過フィルタの出力は高域通過フィルタにおいて2回(第1項と第3項で)使われる。したがって、高域通過フィルタの結果に到達するために、ほかに2つの加算しか行なう必要がない。

【0104】多くのオーバーラップ非最小長可逆フィルタが本発明で用いられてもよい。非オーバーラップ最小

$$1/4 \cdot (1/2 \cdot (-1 - Z^{-1}) + 4 \cdot (Z^{-2} - Z^{-3}) + 1/2 \cdot (Z^{-4} + Z^{-5}))$$

は、無限精度演算によって行なう時には同一である。2番目の式が可逆フィルタを表わしている理由は、整数演算で実際に実施してみれば明白である。低域通過フィルタと高域通過フィルタをハードウェアにより実現した典型例を、図29及び図30に関連して説明する。

【0100】なお、S-変換及びRTS-変換の両方において、低域通過フィルタは入力信号 $x(n)$ のレンジが出力信号 $y_0(n)$ のレンジと同一になるように構成される。例えば、信号が8ビットの画像であるときには、低域通過フィルタの出力も8ビットである。このことは、低域通過フィルタが連続的に適用されるピラミッドシステムにとって重要な特性である。というのは、従来システムでは、出力信号のレンジが入力信号のレンジより大きく、そのことがフィルタの連続的な適用を難しくしていたからである。さらに、低域通過フィルタは2つのタップしか持たないため、このフィルタは非オーバーラップフィルタになる。この特性は、後述のように、フィルタをハードウェアで実現する上で重要である。

【0101】RTS-変換に関し、一実施例では低域通過フィルタ及び高域通過フィルタは次のように定義される。

【0102】

【数20】

長可逆フィルタでフィルタリングを行なう変換システムのフォワード変換とインバース変換の説明を図4に示す。例えば、次に示す種類のフィルタが本発明に用いられてもよい。整数 $L \geq z$ に対し、

【0105】

【数21】

$$y_1(0) = \frac{\sum_{i=0}^{\lfloor L/2 \rfloor} a_i y_0(i) + b d(0) + \sum_{j=\lfloor L/2 \rfloor + 2}^{L-1} c_j y_0(j)}{k}$$

【0106】この高域通過フィルタの長さは $2L$ である。 $L$ が奇数ならば、フィルタはより対称フィルタに近くなる。  $a_i, b, c_i, k$ が整数で、 $k \leq b$ のときは、フィルタは可逆である。 $a_i, b, c_i, k$ が2のべき乗（あるいは、ある2のべき乗の負値もしくは補数）であれば、フィルタの構成が簡単になる。（ $a_i, c_i$ の値と関係なく） $k = b$ ならば、高域通過フィルタ出力  $y_1$  のレンジは最小になる。各  $a_i$  について、 $a_i = -c_i$  と

なる  $c_i$  が丁度 1 つ存在するときには、高域通過フィルタは一定した入力に対し全く応答しない。  $j = (L-1) - i$  の時に  $a_i = -c_i$  ならば、フィルタは、より対称フィルタに近くなろう。

【0107】もう一つの有用な特性が次式で表わされる。

【 0 1 0 8 】

【数 2 2】

$$\sum_{i=0}^{\lfloor L/2 \rfloor} \left[ (a_i)(2i)^m + (a_i)(2i+1)^m \right] + (b)(2(\lfloor L/2 \rfloor + 1))^m - (b)(2(\lfloor L/2 \rfloor + 1) + 1)^m + \sum_{i=\lfloor L/2 \rfloor + 2}^{L-1} \left[ c_i(2i)^m + c_i(2i+1)^m \right] = 0$$

【0109】この特性によつて、高域通過フィルタが、 $m=1$ の時に直線的に変化する入力に対し応答なくなり、 $m=2$ の時に二次曲線的に変化する入力信号に対し応答なくなる、等々となる。ここで $m$ はモーメント条件である。この特性が、RTS-変換のエネルギー集中がS-変換より優れていることの主な根拠である。

【0110】フィルタは可逆性のための最低条件を充足しなければならぬけれども、用途が違えば、他の特性

1	1	-4	16	-16	4	4	-1	-1											
1	1	-3	-3		8	-8	3	3	-1	-1									
-1	-1	0	0		16	-16	0	0	1	1									
-1	-1	4	4		-16	-16	256	-256	16	16									
3	3	-22	-22		128	-128	22	22	-3	-3									

最後のフィルタは (Two/Ten) TT-フィルタであり、3 次増加関数に対して応答しないという特性を有する。なお、 $22=16+2 \times 3$ 、 $3=2+1$ であるので、このフィルタは合計 7 つの加減算によって実現できることに注意されたい。

【0112】フィルタの厳密な可逆性要件は、次のことに留意することにより緩和できる。高域通過係数は、あ

のいずれもを満たさないフィルタ、いくつかを満たすフィルタ、あるいは全部を満たすフィルタを用いてよい。いくつかの実施例においては、次に例示する高域通過フィルタの中の１つが用いられる。本発明を難解にしないため、これらのフィルタは、フィルタの有理型の整数係数を単に表にして表わす形式で下に示す。

【 0 1 1 1 】

る順序で符号化され復号される。前に復号された高域通過係数に対応する画素値が正確に知れば、それを現在の高域通過フィルタリングで使うことができる。例えば、下記のフィルタは、ラスタ順が用いられる時に使用できる。

【0 1 1 3】

【数 2 3】

$$H_1(Z) = \left| \frac{1}{4} \left( \left[ -\frac{1}{2}(1+Z^{-1}) \right] + \left[ \frac{1}{2}(8(Z^2 - Z^{-3}) + (Z^{-4} + Z^{-5})) \right] \right) \right|$$

【0114】単一の固定高域フィルタを使用する必要はない。適応型フィルタを使ってもよいし、あるいは複数のフィルタを使うこともできる。複数のフィルタの適応

化ないし選択のために使われるデータは、ある特定の逆フィルタリング動作に先立って復号器内で入手可能なデータに限定されなければならない。

23

【0115】複数のフィルタを使用する方法は、高域通過係数をプログレッシブ処理することである。1つおきの高域通過フィルタリング演算 ( $y_1(0)$ ,  $y_1(2)$ ,  $y_1(4)$ , ...) は、RTS高域通過フィルタのような可逆フィルタを用いて最初に処理してよい。残りのフィルタリング操作 ( $y_1(1)$ ,  $y_1(3)$ ,  $y_1(5)$ , ...) には、最高6タップの非可逆フィルタを用いてよい。というのは、フィルタのオーバーラップ部分への入力の正確な値が知っているからである。例えば、次に示すフィルタのどれでも使用してよい。

【0116】

```
-1 3 -3 1
-1 4 -4 1
-3 8 -8 3
1 -5 10 -10 5 -1
1 -4 8 -8 4 -1
```

いくつかの実施例では、高域通過フィルタは予測/補間(内挿)操作に置き換えられるかもしれない。予測器/補間器は、特定の予測/補間操作の前に復号器において入手可能なデータを利用し、1対の入力間の差分を予測する。予測した差分は入力の実際の差分から差し引かれ、その差が出力される。一実施例では、DPCM、プログレッシブ符号化あるいは空間領域符号化に使われている従来の予測方法が使用される。

【0117】本発明の低域通過フィルタ及び高域通過フィルタを用いて、マルチ解像度分割が行なわれる。分割レベル数は可変であり任意数でよいが、しかしながら、現在のところ分割数は2乃至5レベルである。

【0118】例えば、可逆ウェーブレット変換が再帰的に1つの画像に適用されるときは、第1レベルの分割は最も細かいディテールもしくは解像度に対し作用する。第1分割レベルでは、画像は4つのサブ画像(すなわちサブバンド)に分割される。各サブバンドは、1つの空間周波数帯域を表わしている。第1レベルのサブバンドはLL0, LH0, HL0, HH0と称される。原画像を分割するプロセスは、水平、垂直両次元における2:1のサブサンプリングを含むので、図5に示されるように、第1レベルのサブバンドLL0, LH0, HL0, HH0はそれぞれ入力を持っている画像の画素数(または係数)の4分の1の数の係数を持っている。

【0119】サブバンドLL0は、水平方向の低い周波数の情報と垂直方向の低い周波数の情報を同時に含んでいる。一般に、画像エネルギーの大部分が当該サブバンドに集中している。サブバンドLH0は、水平方向の低い周波数の情報と垂直方向の高い周波数の情報(例えば水平方向エッジ情報)を含んでいる。サブバンドHL0は、水平方向の高い周波数の情報と垂直方向の低い周波数の情報(例えば垂直方向エッジ情報)を含んでいる。サブバンドHH0は、水平方向の高い周波数の情報と垂直方向の高い周波数の情報(例えばテクスチャもしくはは

24

斜めエッジの情報)を含んでいる。

【0120】この後に続く第2、第3、第4の分割レベルは、前レベルの低周波LLサブバンドを分割することによって作られる。第1レベルの当該サブバンドLL0が分割されることによって、そこそこ精細な第2レベルのサブバンドLL1, LH1, HL1, HH1が作られる(図6)。同様に、サブバンドLL1が分割されることによって、粗い第3レベルのサブバンドLL2, LH2, HL2, HH2が生成される(図7)。また、図8に示されるように、サブバンドLL2が分割されることにより、より粗い第4レベルのサブバンドLL3, LH3, HL3, HH3が作られる。2:1のサブサンプリングにより、第2レベルの各サブバンドは、原画像の16分の1の大きさである。このレベルの各サンプル(つまり画素)は、原画像中の同一位置のそこそこ細いディテールを表現する。同様に、第3レベルの各サブバンドは、原画像の34分の1の大きさである。第3レベルでの各画素は、原画像中の同一位置のかなり粗いディテールを表現する。また、第4レベルの各サブバンドは、原画像の256分の1の大きさである。

【0121】分割された画像はサブサンプリングによる物理的に原画像より小さいので、原画像の格納のために使用されるメモリを利用して、分割サブバンド全部を格納できる。換言すれば、3レベル分割の場合、原画像と分割サブバンドLL0, LL1は捨てられ、保存されない。

【0122】粗いディテールのサブバンド成分と、その次に精細なディテールのサブバンド成分との間には親子の関係が存在する。

【0123】4つのサブバンド分割レベルだけを示したが、個別のシステムの条件に応じて、それ以上のレベルを生成することも可能である。また、DCTのような他の変換あるいは一次元配置のサブバンドによって、異なった親子関係が定義されてもよい。

【0124】マルチ解像度分割のプロセスは、図9のようなフィルタ系列を使って行ない得る。長さLの一次元信号を表わす入力信号は、フィルタユニット401, 402による低域通過フィルタリング及び高域通過フィルタリングの後、ユニット403, 404によって2:1にサブサンプリングされる。ユニット403から出力されるサブサンプル信号は、ユニット405, 406による低域通過フィルタリング及び高域通過フィルタリングの後、ユニット407, 408により2:1にサブサンプリングされる。サブバンド成分L, Hがユニット407, 408の各出力に現われる。同様に、ユニット407の出力信号は、ユニット409, 410により低域通過フィルタリング及び高域通過フィルタリングを施された後、ユニット411, 412によりそれぞれサブサンプリングされる。サブバンド成分L, Hがユニット411, 412の各出力に現われる。上に述べたように、本



25

発明の一実施例においてサブバンド分割に用いられるフィルタは、水平周波数帯域と垂直周波数帯域を低周波帯域と高周波帯域へ分割するためのデジタル直交ミラーフィルタである。

【0125】図10は、二次元の2レベル変換を示している。図11も、図29及び図30に示したような一次元フィルタを使って実現した二次元2レベル変換を示している。一次元フィルタ(451~456)は、サブサンプリングにより不要になる演算を避けるため1画素位置おきに適用される。一実施例では、一次元フィルタは演算を低域通過フィルタ演算と高域通過フィルタ演算との間に割り振る。

【0126】したがって、本発明は、非最小長のオーバーラップ可逆フィルタが使用される圧縮・伸長システムを提供する。図12は、そのようなシステムの一実施例のブロック図である。図12において、最初に階層的分割が行なわれる(ブロック461)。この階層的分割の結果は圧縮器462へ送られて圧縮される。ここで実行される圧縮には、ベクトル量子化、スカラー量子化、ゼロ・ランレングス計数、ハフマン符号化等々が含まれよう。圧縮器462の出力は、オリジナル入力データの圧縮されたものを表すデータからなる。伸長器463は、そのデータを、いつか受け取って伸長することになろう。本発明は、次に、非最小長オーバーラップ可逆フィルタを用い逆分割を行なって(ブロック464)、オリジナルデータの復元データを生成する。

【0127】本発明の可逆ウェーブレットフィルタは、図13に示すように、典型的な分析及び強調システムにも使用し得る。図13において、非最小長オーバーラップ可逆ウェーブレットフィルタを使って、入力データに対し階層的分割が行なわれる(ブロック471、473)。分析ユニット472は、フィルタにより生成された係数を受け取り、それら係数をデジジョンに分類する。すなわち、係数を完全に符号化するのではなく、重要な情報だけが抽出される。例えば、文書保管システムにおいては、空白のページが最も粗い低域通過サブバンドだけを使って認識されよう。もう一つ例を挙げれば、ある特定のサブバンドの高域通過情報だけを用いて、テキスト画像と自然シーン画像とを区別する。階層的分割は、最初に粗いサブバンドによって粗いレジストレーション(registration)が行なわれるような、多数の画像のレジストレーションのためにも使用し得る。もう一つの実施例においては、係数は強調処理またはフィルタリングを施され(ブロック474)、次に逆分割が行なわれる(ブロック475)。鮮明化、エッジ強調、ノイズ制御等々を階層的分割を使って行なうこともできる。このように、本発明は、統合時間/空間領域及び周波数領域の分析並びにフィルタリング/強調処理システム用のウェーブレット変換を提供する。

【0128】<ビット・シグニフィカンス埋め込み(Bi

26

t-Significance Embedded) 符号化>本発明では、ウェーブレット分割の結果として生成される係数がエントロピー符号化される。本発明においては、係数は最初に埋め込み符号化(embedded coding)を施されるが、この符号化では、視覚的に重要な順に係数が順序付けられ、また、より一般的には、何等かの誤差規準(例えば、歪み規準)を考慮して係数が順序付けられる。誤差または歪みの規準には、ピーク誤差と平均2乗誤差(MSE)が含まれる。また、順序付けは、ビット・シグニフィカンス空間配置(bit-significance spatial location)、データベース照会のための妥当性及び方向(垂直、水平、斜め等)を優先させるように行なうことができる。本発明は、複数の埋め込み符号化法を利用し、あるシグニフィカンス・レベルの一部の係数はある符号化法で符号化されるが、残りの係数は別の符号化法で符号化される。本発明においては、周波数ベースモデリングと統合空間/周波数モデリングとが、それぞれ本発明のウェーブレット変換により生成された係数の符号化に利用される異なった2つの埋め込み符号化システムである。周波数ベースモデリングは、低い周波数の1係数を符号化する時に高い周波数の多数の係数を予測する必要がある。統合空間/周波数モデリングは、既知の周波数帯域と近傍画素(またはデータ)を両方利用する。統合空間/周波数モデリングの一例は、本明細書において水平(horizon)モデリングと呼ばれる。

【0129】データはまず符号-絶対値形式にフォーマットされ、次に該データはシグニフィカンス(significance)に基づいて格納される。特定のシグニフィカンス規準を考慮してデータが格納された後、該データは符号化される。周波数ベース符号化及び水平符号化は両方ともビット・シグニフィカンス順序付けをベースにしてよいが、異なるイベント符号化方法を用いる。

【0130】ある信号が $x(n)$ 毎にRビットの精度で表現されているとした場合、本発明の埋め込み符号化は、その信号の各 $x(n)$ の最上位ビット(または複数ビット)を符号化し、次にその下位のビット(または複数ビット)を符号化し、さらに下位のビットというように符号化する。例えば、視覚的に決まる順序付けの場合、中央部分でコーナー沿いまたは縁近傍より高い品質を要求する画像(例えば、ある種の医用画像)は、中央部分の画素の下位ビットが周辺部の画素の上位ビットより先に符号化されるような符号化がなされよう。

【0131】ビット・シグニフィカンス歪み(bit significance distortion)規準に基づいた埋め込みシステムの場合、データのバイナリ値は絶対値によって順序付けられる。値が負でない整数の場合(例えば、画素の輝度に関してそうなる)、採用し得る順序はビットプレーン順(例えば、最上位ビットプレーンから最下位ビットプレーンへの順番)である。2の補数の負整数も許容される実施例では、符号ビットの埋め込み順序は、整数の

27

絶対値の最初の非ゼロのビットと同じである。したがって、1つの非ゼロビットが符号化されるまで、符号ビットは考慮されない。その結果、本発明のビット・シグニフィカンス埋め込みシステムにおいて1つのイベントの可能値は、符号ビットが符号化される前では3元である。この三元イベントは、“非有意 (not significant)”、“積極的有意 (positivesignificant)”、“消極的有意 (negative significant)”である。例えば、符号-絶対値表記法を使うと、-7の16ビット数は  
10000000000000111

である。ビットプレーンベースで、初めの12デシジョンは“非有意”となろう。最初の“1”-ビットは13番目のデシジョンで生じる。13番目のデシジョンは“消極的有意”となろう。符号ビットが符号化された後、その可能なイベントは2つ、すなわち0, 1に減る。14番目と15番目のデシジョンは共に“1”である。

【0132】本発明の一実施例においては、係数を記録するためにリストが用いられる。一実施例では、各係数に関係付けられた1ビットのフラグ（本明細書ではグループフラグと呼ぶ）で、符号ビットが符号化前の係数と、符号ビットが符号化済みの係数を区別する。もう一つの実施例では、フラグビットの代わりに2つ以上のリストを使用できる。別の実施例では単一のリストがフラグ無しで用いられる。

【0133】もう一つの実施例では、リストは使用されない。1つの係数に対する全てのデシジョンが生成され、シグニフィカンスによってラベル付けされた後に、次の係数に対するデシジョンが生成される。こうすることで、リストに係数全部は格納しなくともよくなる。

【0134】＜符号化プロセス及び復号プロセス＞次に説明する図44乃至図55のフローチャートは、本発明の符号化プロセス及び復号プロセスを表わしている。図44及び図45は、本発明の符号化器の変換及びモデリングプロセスを表わすフローチャートである。図44において、符号化器の変換及びモデリングプロセスは、初めに入力データを獲得する（処理ブロック2501）。入力データ獲得後、本発明は可逆ウェーブレットフィルタをかける（処理ブロック2502）。

【0135】次に、別のレベルの分割を要するか判定する（処理ブロック2503）。別のレベルの分割を要するときには、処理ブロック2504へ進み、そこで直前の分割により得られたLL係数に可逆フィルタがかけられ、そして処理ブロック2503へ戻る。このようにして、本発明は任意のレベル数の分割が行なえるようにする。

【0136】別のレベルの分割を要しないときには、処理ブロック2506へ進み、各係数のためのグループフラグをAグループに初期化する。グループフラグを初期化した後、Aパス用のビットプレーン $S_A$ は最上位ビットプレーン（max）とされる（処理ブロック25

28

07）。次にBパス用のビットプレーン $S_B$ はその1つ下位のビットプレーン（max-1）とされる（処理ブロック2508）。次に、Aパス用のビットプレーン $S_A$ を周波数ベースモデルを用いて符号化すべきか否かを判定する（処理ブロック2509）。ビットプレーン $S_A$ を周波数ベースモデルで符号化すべきであれば、処理ブロック2510へ進み、各係数は周波数ベースモデルでモデル化されてからエントロピー符号化される。他方、ビットプレーン $S_A$ が周波数ベースモデルで符号化されるべきでなければ、処理ブロック2511に進み、各係数は統合空間/周波数モデルでモデル化されてからエントロピー符号化される。

【0137】いずれの場合でも、その後処理ブロック2512へ進み、ビットプレーン $S_A$ がゼロ以上であるか判定することにより、当該ビットプレーンが最後のビットプレーンであるか否かを表示する。ビットプレーン $S_A$ がゼロ以上ならば、処理ブロック2509へループバックする。他方、ビットプレーン $S_A$ がゼロ以上でないときには、処理ブロック2513へ進み、ビットプレーン $S_B$ がゼロ以上であるか判定することにより、該ビットプレーンがBパスを受けるべき最後のビットプレーンであるか判定する。ビットプレーン $S_B$ がゼロ以上ならば、処理ブロック2509に進む。しかし、ビットプレーン $S_B$ がゼロ以上でなければ、処理ブロック2514に進み、符号化データがチャンネルへ転送されるかメモリに格納される。符号化データを格納または転送した後、本発明の符号化器の変換及びモデリングプロセスは終了する。

【0138】図46及び図47は、本発明の復号器の変換及びモデリングプロセスを示す。図46において、本発明の復号器の変換及びモデリングプロセスは、まず符号化データの取り込みを行なう（処理ブロック2601）。この符号化データはチャンネルまたはメモリ、あるいは他の伝送システムより受け取られるであろう。符号化データを取り込んだ後に、各係数のグループフラグがAグループに初期化される（処理ブロック2602）。この初期化に続いて、Aパス用ビットプレーン $S_A$ は最上位ビットプレーン（max）とされ（処理ブロック2603）、次にBパス用ビットプレーン $S_B$ はその1つ下位のビットプレーン（max-1）とされる（処理ブロック2604）。そして、各係数の値が初期値のゼロに設定される（処理ブロック2605）。

【0139】各係数の値をゼロに初期化した後、ビットプレーン $S_A$ が周波数ベースモデルを用いて復号されるべきか否かを判定する（処理ブロック2606）。ビットプレーン $S_A$ を周波数ベースモデルで復号すべきであれば、処理ブロック2607へ進み、各係数は周波数ベースモデルでモデル化されたのちエントロピー復号される。ビットプレーン $S_A$ を周波数ベースモデルで復号すべきでないときには、処理ブロック2608へ進み、各

29

係数は統合空間／周波数モデルでモデル化されてからエントロピー復号される。

【0140】各係数がモデル化された後、処理は処理ブロック2609へ進み、ビットプレーン $S_A$ が最後のビットプレーンであるか判定するために、ビットプレーン $S_A$ がゼロ以上であるか調べる。ビットプレーン $S_A$ がゼロ以上ならば、処理ブロック2606に進む。他方、ビットプレーン $S_A$ がゼロ以上でなければ、Bパスのビットプレーン $S_B$ がゼロ以上であるか判定することにより、ビットプレーン $S_B$ がBパス用の最後のビットプレーンであるか否かを表示する(処理ブロック2610)。最後のビットプレーンであるならば、次の復号のために処理ブロック2606へ進む。他方、Bパス用ビットプレーン $S_B$ がゼロ以上でないならば、最も粗い分割レベルによる係数に逆(inverse)可逆フィルタがかけられる(処理ブロック2611)。次に、全てのレベルの逆フィルタ処理が済んだか判定する(処理ブロック2612)。済んでいなければ、残っている最も粗い分割レベルに関する係数に、逆可逆フィルタがかけられる(処理ブロック2613)。そして処理ブロック2612に戻って、全てのレベルの逆フィルタ処理が済んだか再度調べる。

【0141】全レベルが逆フィルタ処理済みとなったならば、処理ブロック2614に進み、復元データの格納または伝送が行なわれる。

【0142】図48及び図49は、各係数のモデリングプロセスの一実施例を示す。図示のプロセスは、周波数ベースモデリングまたはJSFモデリングのモデリングプロセスと符号化または復号を表わしている。すなわち、4つの処理ブロック(2507, 2508, 2607, 2608)のそれぞれは、図48及び図49のモデリングプロセスを用いて実現し得る。図48において、最初のプロセスは、モデリングが1パスで行なわれるべきか否かの判定(処理ブロック2701)で始まる。モデリングが1パスでなされるべきでないときは、ビットプレーン $S_A$ がビットプレーン $S_B$ より大きいと判定する(処理ブロック2702)。大きくなければ、プロセスは処理ブロック2703へ移り、Aパスが実行されるべきでないことを表示するためフラグ(do\_A\_flag)がクリアされる。ビットプレーン $S_A$ がビットプレーン $S_B$ より大きいときは、処理ブロック2704に進み、フラグ(do\_A\_flag)はセットされ、Aパスが実行されることを表示する。

【0143】処理ブロック2703または2704の次に、処理ブロック2705に進み、ビットプレーン $S_B$ がビットプレーン $S_A$ と等しいか判定する。等しくなければ、本発明はBパスを発生させないためにフラグ(do\_B\_flag)をクリアし(処理ブロック2705b)、そして処理ブロック2707に進む。ビットプレーン $S_B$ がビットプレーン $S_A$ と等しいときには、フラ

30

グdo\_B\_flagがセットされてBパスが実行されることを表示し(処理ブロック2706)、処理はまた処理ブロック2707に進む。

【0144】処理ブロック2707で、Aパスフラグがセットされており、ゼロツリー(zerotree)モデリングが行なわれるべきか判定する。フラグが、Aパスが発生しゼロツリーモデリングが行なわれることを表示しているときには、"確定/未確定"(determined/undetermined)フラグが、各係数に対し"未確定"状態に初期化され(処理ブロック2708)、処理ブロック2709へ進む。他方、Aパス表示フラグもゼロツリーモデリング表示もセットされていないときには、処理ブロック2709へ直接進む。処理ブロック2709において、最初の係数が変数Cに設定される。

【0145】最初の係数が変数Cに割り当てられると、Bパス表示フラグがセットされているか判定する(処理ブロック2719)。Bパス表示フラグ(do\_B\_flag)がセットされているならば、本発明は係数Cに対しBパスを実行し(処理ブロック2710)、処理ブロック2711へ進む。他方、Bパスフラグがセットされていないときには、Cに対してBパスは実行されず、処理は処理ブロック2711へ直接進む。

【0146】次に、Aパス表示フラグ(do\_A\_flag)がセットされているか判定する(処理ブロック2711)。Aパス表示フラグがセットされているならば、係数Cに対してAパスが実行される(処理ブロック2717)。その後、処理ブロック2713に進む。Aパス表示フラグがセットされていないときには、係数Cに対するAパスを実行することなく処理ブロック2713へ進む。

【0147】処理ブロック2713において、係数Cが最終の係数であるか判定する。係数Cが最終の係数でなければ、処理ブロック2714に進み、次の係数が変数Cに割り当てられ、処理ブロック2719へ進む。しかし、係数Cが最終の係数であるならば、処理ブロック2715へ進みBパスフラグ(do\_B\_flag)がセットされているか判定する。Bパスフラグがセットされているならば、ビットプレーン $S_{B-1}$ をビットプレーン $S_B$ に設定し(処理ブロック2716)、処理ブロック2717へ進む。Bパス表示フラグがセットされていないならば、処理ブロック2717へ進む。処理ブロック2717では、Aパスフラグがセットされているか判定する。セットされているならば、ビットプレーン $S_{A-1}$ をビットプレーン $S_A$ にトプレーン $S_A$ に設定し(処理ブロック2718)、プロセスは終了する。また、Aパスフラグがセットされていないならば、プロセスは直ちに終了する。

【0148】実施例によっては、ある特定のビットプレーンの係数がAグループであるかBグループであるかを、フラグビットを使わずに判定することができる。

31

これは1係数あたり1ビットのメモリの節約になり、大きな画像の場合には意味があろう。フラグビットを使う代わりに、AND論理を用いて、マスクが係数と比較される。ANDの結果が0ならば、そのビットはAグループであり、そうでなければBグループである。これらマスクの例を8ビットプレーン分につき表7に示す（実施例説明の末尾）。なお、これらマスクは2<sup>(bitplane+1)</sup>の2の補数である（符号ビット含まず）。

【0149】AパスとBパスそれぞれに独自のマスクを割り当てることができるので、Aパスを必要なだけ、対応のBパスの前に実行することができる。17ビットプレーンの場合、3個のAパスが実行され、次に14個のAパスとBパスが同時に実行され、最後に2個のBパスが実行される。普通、AパスのデシジョンのほうがBパスのデシジョンより効率的に符号化できるので、初めに多数のAパスを実行すると、損失性圧縮の質を向上できる。

【0150】図50及び図51は、縮小フラグメモリ（後に詳細説明で述べる）を使用した本発明の符号化器の一実施例を示す。図50において、符号化器の変換及びモデリングプロセスは、初めに入力データを獲得する（処理ブロック2801）。入力データを獲得した後、本発明は可逆ウェーブレットフィルタをかける（処理ブロック2802）。

【0151】次に、別のレベルの分割が必要であるか判定する（処理ブロック2803）。別レベルの分割を要するならば、処理ブロック2804に進み、直前の分割により得られたLL係数に可逆ウェーブレットフィルタがかけられ、そして処理ブロック2803へ進む。このように、本発明によれば任意レベル数の分割を行なうことができる。

【0152】別レベルの分割が必要でないときには、処理ブロック2805へ進み、Aパス用ビットプレーンS<sub>A</sub>は最上位ビットプレーン(max)とされる。そして、Bパス用ビットプレーンS<sub>B</sub>はその次位のビットプレーン(max-1)とされる（処理ブロック2806）。

【0153】次に、マスクM<sub>A</sub>が

【0154】

【数24】

$$-2(S_A+1)$$

【0155】に設定され（処理ブロック2807）、マスクM<sub>B</sub>が

【0156】

【数25】

$$-2(S_B+1)$$

【0157】に設定される（処理ブロック2814）。そして、Aパス用ビットプレーンS<sub>A</sub>を周波数ベースモデルで符号化すべきか否かを判定する（処理ブロック2808）。ビットプレーンS<sub>A</sub>を周波数ベースモデルで

32

符号化すべきであれば、処理ブロック2809に進み、各係数の1ビットが周波数ベースモデルによりモデル化されてエントロピー符号化される。他方、ビットプレーンS<sub>A</sub>が周波数ベースモデルで符号化されるべきでないときには、処理ブロック2810に進み、各係数の1ビットが統合空間/周波数モデルによりモデル化されてエントロピー符号化される。

【0158】いずれの場合にも、次に処理ブロック2811に進み、ビットプレーンS<sub>A</sub>がゼロ以上であるか判定することにより、それが最後のビットプレーンであるか否かを表示する。ビットプレーンS<sub>A</sub>がゼロ以上であるならば、処理ブロック2808へループバックする。他方、ビットプレーンS<sub>A</sub>がゼロ以上でなければ、処理ブロック2812に進み、ビットプレーンS<sub>B</sub>がBパスを施される最後のビットプレーンであるか判断するため、ビットプレーンS<sub>B</sub>がゼロ以上であるか判定する。ビットプレーンS<sub>B</sub>がゼロ以上であれば、処理ブロック2808から処理を繰り返す。しかし、ビットプレーンS<sub>B</sub>がゼロ以上でなければ、処理ブロック2813に進み、符号化データがチャンネルへ転送されるかメモリに格納される。符号化データを格納または転送した後、本発明の符号化器の変換及びモデリングのプロセスは終了する。

【0159】図52及び図53は、縮小フラグメモリを使用した時の本発明の復号器の変換及びモデリングプロセスの別の実施例を示す。図52において、本発明の復号器の変換及びモデリングプロセスは、まず符号化データを取り込む（処理ブロック2901）。この符号化データは、チャンネルかメモリあるいはその他の伝送システムより受け取られるであろう。

【0160】符号化データが受け取られると、Aパス用ビットプレーンS<sub>A</sub>は最上位ビットプレーン(max)とされ（処理ブロック2903）、Bパス用ビットプレーンS<sub>B</sub>はその次位のビットプレーン(max-1)とされる（処理ブロック2904）。各係数の値が初期値のゼロに設定される（処理ブロック2905）。

次にマスクM<sub>B</sub>が

【0161】

【数26】

$$-2(S_B+1)$$

【0162】に設定され（処理ブロック2902）、またマスクM<sub>A</sub>が

【0163】

【数27】

$$-2(S_A+1)$$

【0164】に設定される（処理ブロック2915）。そして、ビットプレーンS<sub>A</sub>が周波数ベースモデルで復号されるべきか否かを判定する（処理ブロック2906）。ビットプレーンS<sub>A</sub>が周波数ベースモデルによって復号されるべきときには、処理ブロック2907へ進

33

み、各係数の1ビットが周波数ベースモデルでモデル化されてエントロピー復号される。ビットプレーン $S_A$ が周波数ベースモデルで復号されるべきでないならば、処理ブロック2908へ進み、各係数の1ビットが統合空間/周波数モデルでモデル化されてエントロピー復号される。

【0165】各係数のモデル化後、処理ブロック2909へ進み、ビットプレーン $S_A$ が最終のビットプレーンであるか判断するため、ビットプレーン $S_A$ がゼロ以上であるか判定する。ビットプレーン $S_A$ がゼロ以上ならば、処理ブロック2902へ進む。他方、ビットプレーン $S_A$ がゼロ以上でないと、B-パスのビットプレーン $S_B$ がゼロ以上であるか判定し（処理ブロック2910）、それによりビットプレーン $S_B$ が最後のB-パス用ビットプレーンであることを表示する。ビットプレーン $S_B$ がゼロ以上ならば、次の復号のため処理ブロック2902へ進む。他方、ビットプレーン $S_B$ がゼロ以上でないときには、最も粗い分割レベルによる係数に対して逆可逆フィルタがかけられる（処理ブロック2911）。次に、全レベルが逆フィルタ処理されたか判定する（処理ブロック2912）。違うときには、残っている最も粗い分割レベルに関する係数に対して逆可逆フィルタがかけられる（処理ブロック2913）。そして処理ブロック2912へ戻り、全レベルの逆フィルタ処理が済んだか判定する。

【0166】全レベルの逆フィルタ処理が済んだならば、処理ブロック2914に進み、復元データの格納または伝送が行なわれる。

【0167】図54及び図55は、各係数のモデリングプロセスの一実施例を示す。なお、図48及び図49と同様、図54及び図55のプロセスは図50及び図51並びに図52及び図53のモデリングステップの実現のために用い得るものである。図54において、最初のプロセスはまず、A-パスが必要で、かつ、 $S_A$ がゼロ以上であるかを判定する（処理ブロック3001）。その通りであるならば、A-パスが実行されるべきであることを表示するフラグ（do\_A\_flag）がセットされ（処理ブロック3004）、処理ブロック3002へ進む。そうでないときには、フラグdo\_A\_flagはクリアされる（処理ブロック3003）。

【0168】ビットプレーン $S_A$ がビットプレーン $S_B$ より大きいときには、処理ブロック3004に進み、A-パスが実行されることを表示するためのフラグがセットされる。ビットプレーン $S_A$ がビットプレーン $S_B$ より大きくないときには、処理ブロック3003へ進み、A-パスが生じることを表示するためのフラグはクリアされる。

【0169】処理ブロック3003または3004に続いて、処理ブロック3002に進み、ビットプレーン $S_B$ がゼロ以上で、かつ、B-パスが必要であるかを判定

34

する。両ビットプレーンが等しくなければ、本発明はフラグ（do\_B\_flag）をクリアしてB-パスを発生させないようにし（処理ブロック3005）、処理ブロック3007へ進む。ビットプレーン $S_B$ がビットプレーン $S_A$ と等しいときには、B-パスが実行されるべきことを表示するためフラグ（do\_B\_flag）がセットされ（処理ブロック3006）、処理ブロック3007へ進む。

【0170】処理ブロック3007において、A-パスフラグがセットされており、かつゼロツリーモデリングが行なわれるべきか判定する。フラグがA-パスが生じるべきで、かつゼロツリーモデリングが実行されるべきであることを表示しているときには、子を有する各係数に対して“確定/未確定”フラグが“未確定”状態に初期化され（処理ブロック3008）、処理は処理ブロック3009へ進む。他方、A-パス表示フラグもゼロツリーモデリング表示もセットされていないときには、処理ブロック3009へ直接進む。処理ブロック3009において、最初の係数が変数Cに設定される。

【0171】最初の係数が変数Cに割り付けられたならばB-パス表示フラグがセットされているか判定する

（処理ブロック3019）。B-パス表示フラグ（do\_B\_flag）がセットされているならば、本発明は係数Cに対してB-パスを実行し（処理ブロック3010）、処理ブロック3011へ進む。他方、B-パスフラグがセットされていないときには、B-パスはCに対し実行されず、処理ブロック3011へ直接進む。

【0172】次にA-パス表示フラグがセットされているか判定する（処理ブロック3011）。A-パス表示フラグがセットされているならば、A-パスが係数Cに対して実行される（処理ブロック3012）。そして、処理ブロック3013へ進む。A-パス表示フラグがセットされていないときには、係数Cに対しA-パスを実行することなく、処理ブロック3013へ進む。

【0173】処理ブロック3013において、係数Cが最終の係数であるか判定する。係数Cが最終の係数でなければ、処理ブロック3014へ進み、次の係数が変数Cに割り付けられ、処理ブロック3019へ進む。しかし、係数Cが最終の係数であるならば、処理ブロック3015へ進み、B-パスフラグ（do\_B\_flag）がセットされているか判定する。B-パスフラグがセットされているならば、ビットプレーン $S_{B-1}$ がビットプレーン $S_B$ とされ（処理ブロック3016）、処理ブロック3017へ進む。B-パス表示フラグがセットされていないときには、処理ブロック3017へ進む。処理ブロック3017では、A-パスフラグがセットされているか判定する。セットされているならば、ビットプレーン $S_{A-1}$ がビットプレーン $S_A$ とされ（処理ブロック3018）、処理は終了する。また、A-パスフラグがセットされていないければ、処理が直ちに終了する。

【0174】＜係数ツリー＞ピラミッドシステムでは、

35

ツリー構造を用いて係数を複数の組みにグループ分けできる。各ツリーのルートは純粋に1つの低域通過係数である。図14は、変換画像の純粋に1つの低域通過係数に関するツリー構造を示す。画像のような二次元信号の場合、ツリーのルートは3個の”子”を持ち、ノード以外のノードはそれぞれ4個の子を持つ。ツリー階層構造は二次元信号に限定されない。例えば、一次元信号の場合、ルートは1個の子を持ち、ルート以外のノードはそれぞれ2個の子を持つ。一次元及び二次元のケースから、より多次元のケースが分かる。

【0175】ツリー構造は、図9乃至図11に示したフィルタの動作からも明らかである。フィルタのペア群の作用とサブサンプリングにより、前述の係数は関係付けられる。

【0176】本発明においては、係数が符号-絶対値形式にされた後、コンテキストモデルが複数の符号化法のどれを、係数をさらに符号化するために用いるべきか判断する。ゼロツリー符号化 (zerotree coding) のような周波数ベース符号化方式は、一定のサブバンド分割に関連した特定の閾値に対する有意データを効率的に符号化する。関連したサブバンド分割の孤立した単一の係数の有意 (singificance) または無意 (insignificance) を表現するシンボルを用いることに加え、全部の子が無意である (全部の子が一定の閾値以下の絶対値を持つ) 無意の親のエントリは一つにまとめられ、一緒に符号化される。これらのツリーはゼロツリー (zerotree) と呼ばれることがある。これらの無意なツリーは、ゼロツリールート (zerotree root) と呼ばれることのある一つの専用シンボルで符号化される。しかしながら、有意な子孫が存在するならば、無意な係数のエントリは”孤立ゼロ” (isolated zero) 用シンボルを用いて符号化される。かくして、1つのツリーは、係数の符号 (sign) が未だ符号化されていないデジションについて、4つのシンボル (積極的有意、消極的有意、孤立ゼロまたはゼロツリールート) を用いて符号化される。

【0177】周波数ベース符号化は、圧縮システムで特に有用である。というのは、無意なツリーの一括符号化が、少数の親係数により多数の子孫係数の無意を予測可能にするからである。ツリー中の子孫係数に関連したエントリはルートから予測できるので、それらの無意を符号化するために余分なシンボルは必要とされない。ツリー全体の無意は非常に低コストで符号化される。よって、上位のビットプレーンは、殆ど、その多くがゼロツリールートでも孤立ゼロでもない (すなわち符号化される必要のない無意なツリーの子である) 無意な係数からできている。

【0178】Shapiroは、Zero tree と呼ばれる周波数ベースモデルを米国特許第5,321,776号で開示している。Shapiroの方法では、主リストと副リストという2つのリストを用いて係数全部

36

を記憶する。シグニフィカンス (significance) レベル毎に、主パスと副パスという2つのパスが作られる。一実施例では、本発明の周波数ベースモデルは Zero tree である。

【0179】別の実施例においては、Zero tree (Shapiroによって述べられた) に類似した周波数ベースモデルが用いられる。複数のリストを使うのではなく、リストが1つだけ用いられ、そのリストの各エレメントは2グループの中の一方向のメンバーである旨マークされる。係数をAグループとBグループに分類することは、Shapiroが主リストと副リストを用いて行なった分類に相当する。Shapiroの複数リストを使う方法は、副リスト中の係数の順序付けのフレキシビリティを大きくできるが、ソフト/ハードの複雑さが増加するという犠牲を伴う。本実施例の単一リスト・ゼロツリー (Single List Zerotree) 法は、AパスとBパスという2つのパスを使う。これら2つのパスはそれぞれ、Shapiroの主パスと副パスに相当する。この単一リスト・ゼロツリーモデルについて、以下に述べる。

【0180】本発明の符号化システムは符号-絶対値形式の係数の1つのリストをメモリ内に作る。このリストの各エレメントは、該エレメントが”AグループまたはBグループ”のいずれのメンバーであるかを表示する1ビットのラベルを持っている。あるステージの初めで、有意であると判明していない係数はAグループに属するとしてラベル付けされる。前のより大きな閾値に対して有意であると前に判明している係数は、Bグループであるとラベル付けされる。リストは、係数を符号化の処理がなされる順に格納する。文字通り最初のステージの開始時点では、どの係数も有意であるとは確定していないので、全ての係数がAグループのメンバーとしてラベル付けされている。係数が有意または無意であると判定された時に、それら係数のエントリが元のAグループ表示からBグループ表示へ変更される。このリストは閾値を次第に細かくして順次更新される。すなわち、リストのマルチパスが生じる。

【0181】一実施例では、Bグループに対応する2元イベントは0次マルコフコンテキストモデルの下でバイナリ算術符号化される。Aグループの係数に対応する4元イベントも、0次マルコフコンテキストモデルの下で符号化される。

【0182】子はその親より先にモデル化されることがないように、本発明のリスト内の係数の順序はツリー構造を保存する。したがって、ツリー構造を保存する、ある順序付けが決まり、それが一貫して使われる。一実施例では、係数は最初に使用された記憶ロケーションより順番にメモリに格納される。別の実施例では、連結リスト (linked list) が用いられるかもしれない。

【0183】一実施例では、係数はビット・シグニフィ

カンスもしくはビットプレーン埋め込み方式で符号化される。係数は最上位のビットプレーンから最下位のビットプレーンへと符号化されるので、データのビットプレーン数を確定しなければならない。本発明では、係数値の絶対値の上限を、データから計算するか、画像の濃さ(depth)とフィルタ係数より得ることによって、ビットプレーン数の確定がなされる。一例を挙げれば、上限が149のときは、8ビットのシグニフィカンスもしくは8つのビットプレーンがある。

【0184】図15及び図16は本発明の単一リスト・ゼロツリー符号化プロセスを示す。一実施例では、図15及び図16に示すプロセスは図48及び図49に示すモデリングプロセスで用いられる。図15において、プロセスは初めに係数Cのグループフラグが“A-グループ”に設定されているか調べる(処理ブロック3221)。“A-グループ”に設定されていないときは、プロセスは終わる。他方、係数Cのグループフラグが“A-グループ”に設定されているときは、処理ブロック3222に進み、C係数の“確定/未確定”フラグが“未確定”に設定されているか判定する。係数Cの“確定/未確定”フラグが“未確定”に設定されていないければ、プロセスは終了する。しかし、係数Cの“確定/未確定”フラグが“未確定”に設定されているならば、処理ブロック3203に進み、係数Cのビット $S_A$ が1であるか判定する。

【0185】係数Cのビット $S_A$ が1でなければ、処理ブロック3207(図16)へ進む。他方、係数Cのビット $S_A$ が1のときには、処理ブロック3204へ進む。係数Cの符号(sign)が正であるか判定する。係数Cの符号が正でないならば、デシジョンは“A-グループ”コンテキストの“消極的有意”として符号化され(処理ブロック3205)、処理ブロック3229へ進む。係数Cの符号が正であるならば、デシジョンは“A-グループ”コンテキストの“積極的有意”として符号化され(処理ブロック3206)、処理ブロック3229へ進む。処理ブロック3229ではCのグループフラグは“B-グループ”に設定される。

【0186】処理ブロック3207で、係数Cの全ての子孫(子)についてビット $S_A$ が0であるか判定する。ビット $S_A$ が0でないときには、デシジョンは“A-グループ”コンテキストの“有意な子のある無意”(01)として符号化され(処理ブロック3208)、プロセスは終了する。他方、係数Cの全部の子孫(子)についてビット $S_A$ が0であるときには、“A-グループ”コンテキストの“ゼロルート”(00)として符号化される(処理ブロック3209)。それから、係数Cの子孫全部に対する“確定/未確定”フラグが“確定”に設定され(処理ブロック3221)、処理は終了する。

【0187】別の実施例では、プロセスの終了判定は、所望の圧縮率に達したか否かでなされる。

【0188】一実施例では、B-パスから得られる2元イベントは、0次マルコフ情報源コンテキストモデルの下でエントロピー符号化される。A-パスより得られる2ビットのアルファベット(サイズは4)も、0次マルコフ情報源コンテキストモデル下で、4元(サイズ4のアルファベット)算術符号化器により符号化される。図17及び図18は、縮小フラグメモリを使う本発明の単一リストゼロツリー符号化プロセスの別の実施例を示す。一実施例では、図17及び図18のプロセスが図54及び図55のプロセスにおけるA-パスとして使用される。図17において、プロセスは初めに係数Cとマスク $M_A$ とを論理積した結果が0であるか判定する(処理ブロック3201)。0でなければ、プロセスは終了する。他方、係数Cをマスク $M_A$ と論理積した結果が0ならば、処理ブロック3202に進み、係数Cの親の“確定/未確定”フラグが“未確定”に設定されているか判定する。係数Cの親の“確定/未確定”フラグが“未確定”に設定されていないときは、プロセスは終了する。しかし、係数Cの親の“確定/未確定”フラグが“未確定”に設定されているときには、処理ブロック3203に進み、係数Cのビット $S_A$ が1であるか判定する。

【0189】係数Cのビット $S_A$ が1でなければ、処理ブロック3207へ進む。他方、係数Cのビット $S_A$ が1であるならば、処理ブロック3204に進み、係数Cの符号(sign)が正であるか判定する。係数Cの符号が正でなければ、デシジョンは“A-グループ”コンテキストの“消極的有意”で符号化され(処理ブロック3205)、プロセスは終了する。係数Cの符号が正であるならば、デシジョンは“A-グループ”コンテキストの“積極的有意”として符号化され(処理ブロック3206)、プロセスは終了する。一実施例では、4元符号化器が用いられ、4元デシジョンが1つのコンテキストで符号化される。別の実施例では、バイナリ符号化器が使用され、3つのコンテキストが用いられる(例えば、この3つのコンテキストは、デシジョンの第1ビット、第1ビットが0のときの第2ビット、及び第1ビットが1のときの第2ビット、である)。

【0190】処理ブロック3207では、係数Cの子孫(子)全てにつきビット $S_A$ が0であるか判定する。ビット $S_A$ が0でなければ、デシジョンは“A-グループ”コンテキストの“有意の子がある無意”、“孤立ゼロ”(01)として符号化され(処理ブロック3208)、プロセスは終了する。他方、係数Cの子孫(子)全てにつきビット $S_A$ が0であるならば、デシジョンは“A-グループ”コンテキストの“ゼロツリー・ルート”(00)として符号化される(処理ブロック3209)。そして、係数Cの“確定/未確定”フラグが“確定”に設定される(処理ブロック3210)。続いて、係数の子孫(同様に子孫を持つ)の全部に対する“確定/未確定”フラグが“確定”に設定され(処理ブロック



39

3211)、プロセスは終わる。

【0191】<復号ステップ>本発明においては、復号は符号化と密接行進的に行なわれる。

【0192】図19はゼロツリー水平符号化(zerotree horizon coding)プロセスのためのA-パスプロセスの一実施例を示し、これは図48及び図49のプロセスと共に用い得る。図19において、プロセスは初めに係数Cのグループフラグが“A-グループ”に設定されているか判定する(処理ブロック3521)。“A-グループ”に設定されていないければ、プロセスは終了する。しかし、“A-グループ”に設定されているならば、処理ブロック3528に進み、係数Cの“確定/未確定”フラグが“未確定”に設定されているか判定する。“未確定”に設定されていないければ、プロセスは終了する。“未確定”に設定されているならば、処理ブロック3502に進み、3元デシジョンはA-グループコンテキストで復号される。

【0193】次に、デシジョンが“積極的有意”であるか判定する(処理ブロック3503)。デシジョンが“積極的有意”であるならば、係数の符号が正に設定され(処理ブロック3505)、係数の絶対値が

【0194】

【数28】

$$2^S A$$

【0195】に設定され(処理ブロック3507)、係数Cのグループフラグが“B-グループ”に設定され(処理ブロック3541)、処理は終了する。

【0196】デシジョンが“積極的有意”でないときは(処理ブロック3503)、デシジョンが“消極的有意”であるか判定する(処理ブロック3504)。デシジョンが“消極的有意”でないときには、処理ブロック3510に進み、デシジョンが“ゼロツリー・ルート”であるか判定する。デシジョンが“ゼロツリー・ルート”でなければ、プロセスは終わる。デシジョンが“ゼロツリー・ルート”ならば、係数Cの子孫全部の“確定/未確定”フラグが“未確定”に設定され(処理ブロック3531)、プロセスは終了する。

【0197】しかし、処理ブロック3504の判定でデシジョンが“消極的有意”と判断されたときには、係数Cの符号が負に設定され(処理ブロック3506)、係数Cの絶対値が

【0198】

【数29】

$$2^S A$$

【0199】に設定され(処理ブロック3507)、係数CのグループフラグがB-グループに設定され(処理ブロック3541)、プロセスは終了する。

【0200】図20は縮小フラグメモリを用いるゼロツ

40

リー水平復号プロセスのためのA-パスプロセスの他の実施例を示し、これは図54及び図55に述べたプロセスにおいて用いることができる。図20において、プロセスは初めに係数Cとマスク $M_A$ の論理積結果が0であるか判定する(処理ブロック3501)。0でなければ、プロセスは終了する。しかし、係数Cとマスク $M_A$ を論理積した結果が0ならば、処理ブロック3508へ進み、係数の“確定/未確定”フラグが“未確定”であるか判定する。“未確定”でなければ、処理は終了する。“未確定”であれば、処理ブロック3502に進み、3元デシジョンはA-グループコンテキストで復号される。

【0201】次に、デシジョンが“積極的有意”であるか判定する(処理ブロック3503)。デシジョンが“積極的有意”であるならば、係数の符号が正に設定され(処理ブロック3505)、係数の絶対値が

【0202】

【数30】

$$2^S A$$

【0203】に設定され(処理ブロック3507)、プロセスは終了する。

【0204】デシジョンが“積極的有意”でなければ、それが“消極的有意”であるか判定する(処理ブロック3504)。“消極的有意”でなければ、処理ブロック3509に進み、デシジョンが“ゼロツリー・ルート”であるか判定する。デシジョンがゼロツリー・ルートでなければ、プロセスは終了する。デシジョンがゼロツリー・ルートであるならば、係数Cの“確定/未確定”フラグが“確定”に設定され(処理ブロック3510)、係数Cの全子孫(これも同様に子孫を持つ)に対する“確定/未確定”フラグが“確定”に設定され(処理ブロック3511)、プロセスは終了する。

【0205】しかしながら、処理ブロック3504のテストでデシジョンが“消極的有意”であると判定したときには、係数Cの符号は負に設定され(処理ブロック3506)、係数Cの絶対値は

【0206】

【数31】

$$2^S A$$

【0207】に設定され(処理ブロック3507)、プロセスは終了する。

【0208】Shapiroは4元デシジョンを使ってツリーを記述する方法を選択したが、その代案が存在する。ツリーのルートを符号化する時にツリー全体の特性をより詳細に記述するため、拡大したアルファベットを使用してもよい。一実施例では、以下の6元デシジョンの集合が用いられる。

【0209】・無意の子を持つ無意(ゼロツリー・ルー



41

ト)

- ・少なくとも1つの有意な子を持つ無意
- ・有意、積極的かつ全てのが非消極的 (non-negative)
- ・有意、積極的かつ少なくとも1つの子が消極的
- ・有意、消極的かつ全ての子が非積極的 (non-positiv e)

- ・有意、消極的かつ少なくとも1つの子が積極的

この実施例では、ツリー全体について、無意に加え符号 (sign) 情報も予測される。別の実施例では、他の符号 (sign) 制限または絶対値制限のあるツリーが予測可能である。択一的予測器が、テクスチャあるいはマルチ解像度のフィーチャを表現する場合に特に有効かもしれない。アルファベットの拡大とともに、高次のマルコフコンテキスト (後述) を使用するのが有効かもしれない。

【0210】<マルチパスリストベース (Multipass List Based) 統合空間/周波数埋め込みモデリング>本発明において、本明細書に開示された水平順序 (horizon order) モデリングのような周波数埋め込み符号化 (frequency embedded coding) は、A-グループの係数に対応する3元イベントを符号化する。水平符号化では、符号化ステップに先行する初期化は全て、周波数ベースのシステムと同一である。一実施例では、バイナリエントロピー符号化は、“A-グループ 絶対値”、“A-グループ 符号 (sign)”、及び“B-グループ” という3つのコンテキストで行なわれる。

【0211】図21は本発明の単一リスト水平符号化プロセスのためのA-パスの一実施例のフローチャートである。このプロセスは図48及び図49のプロセスで利用し得る。図21において、このA-パスプロセスは初めに係数Cのグループフラグが“A-グループ”であるか判定する (処理ブロック3111)。“A-グループ”でなければ、プロセスは終了する。係数Cのグループフラグが“A-グループ”に設定されているときは、処理ブロック3102に進み係数Cのビット $S_A$ が1であるか判定する。係数Cのビット $S_A$ が1でなければ、デシジョンは“A-グループ”コンテキストの無意

(0)として符号化され (処理ブロック3103)、プロセスは終了する。係数Cのビット $S_A$ が1ならば、処理ブロック3104へ進み係数Cの符号 (sign) が正であるか判定する。係数Cの符号が正ならば、デシジョンは“A-グループ”コンテキストの“積極的有意”(10)として符号化され (処理ブロック3106)、処理ブロック3117へ進む。他方、係数Cの符号が正でなければ、デシジョンは“A-グループ”コンテキストの“消極的有意”(11)として符号化され (処理ブロック3105)、処理ブロック3117へ進む。処理ブロック3117で、係数Cのグループフラグが“B-グループ”に設定される。

【0212】図22は、縮小フラグメモリを使用する本発明の単一リスト水平符号化プロセスのためのA-パス

42

の他の実施例のフローチャートである。このプロセスは図54及び図55のプロセスで用い得る。図22において、このA-パスプロセスは初めに係数Cをマスク $M_A$ と論理積した結果が0であるか判定する (処理ブロック3101)。0でなければ、プロセスは終了する。係数Cのマスク $M_A$ の論理積結果が0であるときは、処理ブロック3102に進み係数Cの $S_A$ が1であるか判定する。係数Cの $S_A$ が1でなければ、デシジョンは“A-グループ”コンテキストの無意(0)として符号化され (処理ブロック3108)、プロセスは終了する。係数Cの $S_A$ が1であるならば、処理ブロック3104へ進み係数Cの符号 (sign) が正であるか判定する。係数Cの符号が正であるならば、デシジョンは“A-グループ”コンテキストの“積極的有意”(10)として符号化され、プロセスは終了する。他方、係数Cの符号が正でなければ、デシジョンは“A-グループ”コンテキストの“消極的有意”(11)として符号化され (処理ブロック3105)、プロセスは終了する。

【0213】<復号ステップ>図23は本発明の単一リスト水平復号プロセスのためのA-パスプロセスの一実施例を示し、図48及び図49のプロセスに利用し得る。図23において、このプロセスは初めに係数Cのグループフラグが“A-グループ”に設定されているか判定する (処理ブロック3411)。“A-グループ”に設定されていないならば、プロセスは終了する。しかし、係数Cのグループフラグが“A-グループ”に設定されているならば、処理ブロック3402に進み3元デシジョンはA-グループコンテキストで復号される。

【0214】次に、デシジョンが“積極的有意”であるか判定する (処理ブロック3403)。デシジョンが“積極的有意”であるならば、係数Cの符号 (sign) が正に設定され (処理ブロック3405)、係数の絶対値が

【0215】

【数32】

$$2^{S_A}$$

【0216】に設定され (処理ブロック3407)、係数Cのグループフラグが“B-グループ”に設定され (処理ブロック3418)、プロセスは終了する。

【0217】デシジョンが“積極的有意”でなければ、デシジョンが“消極的有意”であるか判定する (処理ブロック3404)。デシジョンが“消極的有意”でなければ、プロセスは終了する。しかし、デシジョンが“消極的有意”であるならば、係数Cの符号が負に設定され (処理ブロック3406)、係数Cの絶対値が

【0218】

【数33】

$$2^{S_A}$$

【0219】に設定され (処理ブロック3407)、係

43

数Cのグループフラグが”B-グループ”に設定され  
(処理ブロック3418)、プロセスは終了する。

【0220】図24は縮小フラグメモリを使用する単一リスト水平復号プロセスのためのA-パスプロセスの他の実施例を示し、これは図54及び図55のプロセスにおいて用い得る。図24において、プロセスは初めに係数CとマスクM<sub>A</sub>の論理積結果が0であるか判定する

(処理ブロック3401)。0でなければ、プロセスは終了する。しかし、係数CとマスクM<sub>A</sub>の論理積結果が0ならば、処理ブロック3402に進み3元デシジョンはA-グループコンテキストで復号される。

【0221】次にデシジョンが”積極的有意”であるか判定する(処理ブロック3403)。デシジョンが”積極的有意”ならば、係数Cの符号(sign)が正に設定され(処理ブロック3405)、係数の絶対値が

【0222】

【数34】

$$2^{S_A}$$

【0223】に設定され(処理ブロック3407)、プロセスは終了する。

【0224】デシジョンが”積極的有意”でなければ、デシジョンが”消極的有意”であるか判定する(処理ブロック3404)。デシジョンが”消極的有意”でなければプロセスは終了する。しかし、デシジョンが”消極的有意”であるならば、係数Cの符号(sign)が負に設定され(処理ブロック3406)、係数Cの絶対値が

【0225】

【数35】

$$2^{S_A}$$

【0226】に設定され(処理ブロック3407)、プロセスは終わる。

【0227】<ゼロツリー符号化/復号及び水平符号化/復号のためのB-パス>一実施例においては、本発明のゼロツリー符号化/復号と水平符号化/復号のためのB-パスプロセスは同一である。符号化プロセスと復号プロセスのためのB-パスアルゴリズムの実施例が図25及び図26と図27及び図28にそれぞれ示されている。

【0228】図25はゼロツリー符号化及び単一リスト水平符号化のプロセスのために部分的に用いられるB-パスプロセスの一実施例を示し、図48及び図49のプロセスに利用し得る。図25において、プロセスは最初に係数Cのグループフラグが”B-グループ”に設定されているか判定する(処理ブロック3311)。”B-グループ”に設定されていないければ、プロセスは終了する。他方、グループフラグが”B-グループ”に設定されているならば、処理ブロック3302に進み係数CのビットS<sub>B</sub>が”1”であるか判定する。係数Cのビット

44

S<sub>B</sub>が”1”でなければ、デシジョンは”B-グループ”コンテキストの”0”として符号化され(処理ブロック3303)、プロセスは終了する。係数CのビットS<sub>B</sub>が”1”であるならば、デシジョンは”B-グループ”コンテキストの”1”として符号化され(処理ブロック3304)、プロセスは終了する。

【0229】図26はゼロツリー符号化及び単一リスト水平符号化のプロセスに部分的に用いられる、縮小フラグメモリを使うB-パスプロセスの他の実施例を示し、図54及び図55のプロセスに使用し得る。図26において、プロセスは最初にマスクM<sub>B</sub>と係数Cの論理積結果が0でないか判定する(処理ブロック3301)。0であるならば、プロセスは終了する。他方、マスクM<sub>B</sub>と係数Cの論理積結果が0でなければ、処理ブロック3302に進み係数CのビットS<sub>B</sub>が”1”であるか判定する。係数CのビットS<sub>B</sub>が”1”でなければ、デシジョンは”B-グループ”コンテキストの”0”として符号化され(処理ブロック3303)、プロセスは終了する。係数CのビットS<sub>B</sub>が”1”であるならば、デシジョンは”B-グループ”コンテキストの”1”として符号化され(処理ブロック3304)、プロセスは終了する。

【0230】図27は本発明のB-パス復号の一実施例を示し、図48及び図49のプロセスに利用し得る。図27において、最初に係数Cのグループフラグが”B-グループ”に設定されているか判定する(処理ブロック3611)。そのように設定されていないければ、プロセスは終了する。しかし、係数Cのグループフラグが”B-グループ”に設定されているならば、デシジョンは”B-グループ”コンテキストで復号される(処理ブロック3602)。次に、デシジョンが”1”であるか判定する(処理ブロック3603)。デシジョンが”1”でなければプロセスは終了する。デシジョンが”1”ならば、係数CのビットS<sub>B</sub>がセットされ(処理ブロック3604)、プロセスは終了する。

【0231】図28は縮小フラグメモリを使用する本発明のB-パス復号の他の実施例を示し、図54及び図55のプロセスにおいて使用できる。図28において、最初に係数CとマスクM<sub>B</sub>との論理積の結果が非0か判定する(処理ブロック3601)。係数CとマスクM<sub>B</sub>との論理積の結果が0ならば、プロセスは終了する。しかし、係数CをマスクM<sub>B</sub>と論理積した結果が非0ならば、デシジョンは”B-グループ”コンテキストで復号される(処理ブロック3602)。そして、デシジョンが”1”であるか判定する(処理ブロック3603)。デシジョンが”1”でなければ、プロセスは終了する。デシジョンが”1”であれば、係数CのビットS<sub>B</sub>がセットされ(処理ブロック3604)、プロセスは終了する。

【0232】本発明は、ゼロツリー順序化と水平順序

45

符号化の組合せを用いて、可逆ウェーブレットにより生成された係数のビット・シグニフィカンス符号化を行なう。なお、A-グループとB-グループの両方並びに“A”パスと“B”パスに対応する3元と2元のイベントを使うことは、ゼロツリー順序付けの利用から水平順序付けの利用への切り替えが任意のAパスの終わりでもなされるという点で、非常に重要である。これは、下位ビットでのゼロツリー順序付けに伴う予測の非効率を補う。したがって、本発明においては、システムは初めに上位ビットデータをゼロツリー符号化し、リストのある回数のパスの後に、つまり、いくつかのビットプレーンが符号化された後に、本発明の符号化器は水平符号化によって残りのデータを符号化するように切り替わる。そのパスの回数は、統計的に選んでもよいし、あるいは、ゼロツリー順序付け符号化ブロックの性能をモニタして適応的に選んでもよい。

【0233】<コンテキストモデルの例案>一実施例では、5個のバイナリコンテキストビン(bin)が用いられる。これは、1024個より若干多いコンテキストを用いるJBIGのような他のシステムに比べると少ない。もっと多くのコンテキストを用いれば、圧縮率は向上するであろう。デシジョンは、空間的位置、レベル、及び/またはビット位置により条件付けられてもよい。デシジョンは、現在データと空間的位置、レベル及び/またはビット位置が接近した、前に符号化されたデータにより条件付けられてもよい。一般的に、前述の0次マルコフコンテキストは、より高次のマルコフコンテキストによって置き換えてもよい。

【0234】いくつか例を挙げるならば、次のとおりである。各仮数(いくつかの実施例におけるB-グループデータ)の最上位の(したがって最も容易に予測される)ビットは、その他のビットとは異なったコンテキストを用いてもよい。有意/非有意のデシジョンは、同一変換レベルの空間的に接近した前の係数に対して下された同様のデシジョンにより条件付けられてもよい。同様に、有意な係数の符号ビットは、同一レベルの空間的に接近した前の係数の符号ビット、あるいは親の係数の符号ビットにより条件付けられてもよい。

【0235】コンテキストモデルの改良は、空間的構造またはマルチ解像度構造を有する画像を圧縮する場合に特に重要である。線図形またはテキストの濃淡画像は、そのような種類の構造の両方を持つ画像の一例である。コンテキストモデルの改良は、ある指定されたピーク誤差で圧縮及び伸長されなければならないファイルの圧縮にも重要である。

【0236】<他の実施例>本発明は、ハードウェア及び/またはソフトウェアで実現し得る。ハードウェアにより本発明を実施するには、ウェーブレットフィルタ、同フィルタにデータを供給するためのメモリ/データフロー管理機能、本発明の埋め込み符号化を制御するため

46

のコンテキストモデル、同コンテキストモデルにデータを提供するためのメモリ/データフロー管理機能、及びバイナリエントロピー符号化器を実現する必要がある。

【0237】<ウェーブレットフィルタ>本発明のフォワード・ウェーブレットフィルタの一実施例が図29に示されている。図29に示したウェーブレットフィルタは、 $x(2) \sim x(5)$ として示した16ビットの2の補数の入力画素を4個取り込む。

【0238】図29において、2タップ”11”低域通過フィルタは1個の16ビット加算器1001を用いる。その出力はそれぞれSとDと呼ばれる。この加算器の出力(S)は、1ビットシフトブロック1003により16ビットに丸められる。この1ビットシフトブロック1003は、17ビット入力を1ビット右シフトすることにより2で割る働きをする。

【0239】6タップ”-1 -1 8 -8 1 1”高域通過フィルタは、 $-S_0 + 4D_1 + S_2$ という計算を必要とする。 $S_2 - S_0$ の計算は、1ビットシフトブロック1003の出力及び $Y_0(0)$ を受け取る16ビット減算器1005により求められる。 $4D_1$ 項は、減算器1002と2ビットシフトブロック1004を用いて計算される。16ビット減算器1002の出力は、2ビット左シフトされることにより4倍される。2ビットシフトブロック1004の $4D_1$ 出力と減算器1005の出力の加算が20ビット加算器1006により行なわれる。この最後の加算器の出力は、2ビットシフトブロック1007によって18ビットに丸められる。この2ビットシフトブロック1007は、その20ビット入力を2ビット右シフトすることにより、4で割る働きをする。

【0240】このように、必要とされる演算ハードウェア全体(中間結果を格納するためのレジスタは数に依らない)は、

- ・16ビット加算器 1個
- ・16ビット減算器 2個
- ・19ビット加算器 1個

である。なお、シフトは配線により行なわれるので論理は不要である。

【0241】他の実施例では、Nビットサイズの入力の場合、1個のNビット加算器、2個のNビット減算器及び1個の(N+3)ビット加算器が使われることになる。

【0242】これらの加算器/減算器はハードウェアコストが非常に低いので、希望するならばフィルタの並列構成も利用できる。

【0243】あるいは、 $x(3)$ と $x(2)$ の減算の代わりに、 $x(4) - x(5)$ を計算し、これを次のシフトまたはフィルタ処理のための $x(2) - x(3)$ として必要になるまでセーブしてもよい。このフォワードフィルタ(及び下に述べる逆フィルタ)のいずれも、より高いスループットを得るためパイプライン化してもよい。

47

【0244】逆ウェーブレットフィルタが図30に示されている。 $Y_0(0)$ 、 $Y_0(2)$ 入力は減算器1101により減算される。この減算の結果は2ビットシフトブロック1102で右に2ビットシフトされる。これは事実上、減算器1101の出力を4で割算することである。この2ビットシフトブロック1102の出力と入力 $Y_1(0)$ との間で減算が行なわれる。入力 $Y_0(1)$ は、1ビットシフトブロック1103により左に1ビットシフトされることにより、2倍される。 $Y_0(1)$ が1ビットシフト(2倍)された後、そのシフト後の値のLSBは減算器1104の出力から得られたLSBであり、1ビットシフトブロック1103の16ビット出力と結合されて加算器1105と減算器1106の入力となる。加算器1105及び減算器1106のもう一方の入力は、減算器1104の出力である。加算器1105及び減算器1106の出力はその後、クリッピングを施される。

【0245】2つのクリップ操作の1つを選んで使用できる。いずれの場合でも、20ビットの値が1ビットシフトされ(2で割られ)、19ビット値とされる。非損失性圧縮だけを実行するシステムの場合、下位16ビットを出力してよい(残り3ビットは無視してよい)。損失性システム(または損失性/非損失性システム)では、19ビット値は、負ならば0に設定され、 $2^{16}-1$ を超えときには $2^{16}-1$ に設定され、それ以外ならば下位16ビットを出力してよい。

【0246】Nビットサイズの入力の場合、1個のNビット減算器、1個の(N+2)ビット減算器、1個の(N+3)ビット加算器及び1個の(N+3)ビット減算器が使われることになる。そして、クリップユニットはNビットを出力する。

【0247】<メモリの使い方>本発明のウェーブレットフィルタのためのメモリ及びデータフロー管理に関しては、1枚のフルフレームがメモリにぴったり入る画像の場合、メモリ/データフロー管理は難しい問題ではない。1024×1024×16ビットの医用画像(すなわちサイズが2Mバイト)の場合でさえも、1つのフルフレームバッファを要求することは多くの用途にとって合理的である。より大きい画像(例えば、A4、400DPI、4色の画像は約50Mバイトの大きさである)については、限られた量のラインバッファメモリを用いてウェーブレット変換を行なうのが望ましい。

【0248】なお、本発明が1パスシステムを実現するのにフルフレームバッファは必要でない。このため、必要とされるメモリは100分のいくつになる(大きな画像のためのフルフレームバッファを使用する場合に比べ)。本発明の1パスシステムは後述する。

【0249】フィルタメモリに格納されるデータは、埋め込み符号化及びバイナリエントロピー符号化される係数の系列である。埋め込み符号化は、周波数ベース符号化または水平符号化の利用を調整し、かつデータを適当

48

な順序で供給するため、あるコンテキストモデルを用いる。このコンテキストモデルは、メモリ管理手順と関連して動作する。フルフレームメモリを持つシステムでは、データを適当な順序で供給することは難しくない。フルフレームバッファを持たないシステムの場合、本発明の1パスの実施例の変換データ管理手順(後述)は、コンテキストモデルが1つのツリー分の係数だけバッファすればよいように係数をコンテキストモデルに供給する。1パスの周波数ベースコンテキストモデル及び1パスの統合空間/周波数コンテキストモデルは、一度に1つのツリーを処理する。

【0250】本発明の埋め込み操作の結果は、本発明の周波数ベースモデリング機構及び本発明の統合空間/周波数モデリング機構よりビットストリームを生成させる。これらのビットストリームは、次にバイナリエントロピー符号化器により符号化される。

【0251】フルフレームバッファを持つシステムでは、どのようなバイナリエントロピー符号化器(あるいは他の適当な符号化器)を用いてもよい。フルフレームバッファを持たないシステムでは、複数の独立した符号化器を使用するか、あるいは1つの符号化器が複数の独立した符号化器をシミュレートできるか、のいずれかでなければならない。また、独立した複数の符号化器の出力を管理するためにメモリまたはチャンネルの管理が必要となる。本発明の利点は、管理されるべきデータが優先順位付けされる(埋め込まれる)ことである。圧縮中または伝送中に十分なスペースまたは帯域幅を利用できないときには、重要度の低いデータを直ちに廃棄することができるため、合理的な損失性圧縮が可能である。

【0252】<本発明の1パスシステム>本発明は、システムの入力データを、それを受け取った時に完全に処理できる1パス変換を提供する。このようなシステムでは、データの処理が後続データに依存しない。画像を圧縮するために必要とされるメモリが、画像の長さに依存しない。本発明は、かかる依存性の排除により、データ全部の処理が完了する前に圧縮データを出力可能なシステムを提供する。

【0253】<A. 1パス変換のためのデータ管理>図31は本発明の教えるところに従って帯状にラスタ順に圧縮中の画像の一部を示す。4レベルの分割を想定する。各ツリーは $2^4 \times 2^4 = 16 \times 16 = 256$ 個の係数を有する。しかし、本発明におけるウェーブレット変換の高域通過フィルタはオーバーラップしているので、各ツリーは256個より多数の入力画素に依存する。2タップ”11”低域通過フィルタ(L)は全くオーバーラップを生じず、オーバーラップは全て6タップ”-1 -1 -8 -8 1 1”高域通過フィルタ(H)に起因する。低域通過フィルタを3回、続いて高域通過フィルタを1回、直列的にかける場合(LL LH)に、最大のオーバーラップが生じる。低域通過フィルタを3回かける(LL

49

L)には、 $2^3=8$ 個の入力画素のサポートを必要とする。8×8画素サイズのサポート領域が図31に示されている。高域通過フィルタが直列に入れられた時には、サポート領域は $(6 \times 2^3) \times (6 \times 2^3) = 48 \times 48$ 画素である。1つの48×48画素のサポート領域は、図31に見られるように36個の8×8ブロックからなる。

【0254】図31に示した48×48画素サポート領域の係数が現在処理されているとする。このサポート領域の淡く陰を付けた部分は、前のサポート領域で既に利用された画素を表わす。サポート領域の外側の淡く陰を付けた部分は、前サポート領域で既に利用され、かつ、これから先のサポート領域で必要となる画素を示している。黒い16×16領域は、当該サポート領域のまだ利用されていない画素を含む部分である。同様に、濃く陰を付けた16×16領域は、まだ利用されておらず、次の48×48サポート領域で利用されることになる画素を含む。1個の3レベル16×16変換が計算され、他の8個の3レベル16×16変換についての前の結果がバッファから読み出され、そして変換の第4レベルがこれら9個の3レベル16×16変換に施される。この実行に要するバッファリングは、 $(2 \times \text{画像幅} + 32) \times 16$ 画素に対する3レベル変換係数を格納するに足り、その上に16ライン(1つの帯)バッファ分の画素を格納するに足りるということである。

【0255】図32は1パスウェーブレットフィルタリングユニットの一実施例のブロック図であり、このユニットはフィルタ制御ユニット1301、メモリ1302及びフィルタ1303からなる。フィルタ1303は図29に関連して説明したフィルタからなる。メモリ1302は図31に関連して上に述べたメモリを指し、画素または係数を格納する。フィルタ制御ユニット1301は、メモリ1302とフィルタ1303の間のデータフローを決定する。フィルタ制御ユニット1301の動作は後述する。

【0256】図33は別のウェーブレットフィルタユニットを示す。高速動作を達成するため、複数のフィルタを使用可能である。一実施例では、フィルタ1303は4個または5個の入力を必要とし(例えば、逆フィルタ、フowardフィルタ)、2個の出力を出すので、必要とされるメモリ帯域幅はかなりになる。メモリは、1ロケーションあたり複数の画素/係数を、また複数のバンク及び/または複数のポートを持つかもしれない。メモリインターフェイスユニット1401は、処理中に利用されるローカルデータ用の小バッファを提供することによって、必要とされるメモリ帯域幅を減らす。メモリインターフェイスユニット1401はまた、メモリ1302の入出力(I/O)とフィルタのI/Oとの間のマルチプレクス/デマルチプレクスを行なう。

【0257】フィルタリングのために必要とされるメモ

50

リ帯域幅に加え、メモリ1302への画素入力及びコンテキストモデルへの係数の出力のために余分な帯域幅が必要となるかもしれない。画素がラスタ順に入力されるならば、帯バッファのために余分なメモリが必要になる。

【0258】メモリが1ロケーションあたり複数の要素(画素または係数)を格納する場合、水平方向または垂直方向に隣接する要素を1行または1列に格納するのではなく、1個のN×Nブロック(Nは2のべき乗)の要素が同じロケーションを共有すれば、必要なメモリアクセス及びバッファの量が減少しよい。こうすると、垂直方向アクセス及び水平方向アクセスのいずれにも等しく便利である。

【0259】図34に示されるように、複数バンクのメモリも、それを水平方向アクセスと垂直方向アクセスで均等に利用できるように構成し得る。2バンクの場合、どちらかのバンクを選択するために用意される1ビットのバンク選択ビットが、例えば、垂直座標と水平座標の最下位ビットを論理和することによって作られるかもしれない。4バンクの場合、2ビットのバンク選択ビットが、水平座標と垂直座標の最下位2ビットを加算(2ビット加算器を用いてモジュロ4で)することによって作られるかもしれない。

【0260】図35は、フィルタ制御ユニット1301(図32)による2レベル分割を実現するための1パスフィルタ動作を示す。なお、説明の都合上、まず2レベル分割について論じてから、本発明の一般的手法を説明する。他の実施例では、3レベル、4レベル、またはそれ以上のレベル数の分割が用いられる。2レベル分割は、1ツリーあたり16個の係数を持ち、前に利用されなかった16個の入力画素について計算を必要とする。1ツリーの係数に対するフィルタリングは、入出力レートと調和すべく16時間単位以内で成し遂げられる。この例では、必要とされるスループットである1単位時間あたり2フィルタリング動作を達成するため、並列動作する2個のフィルタが用いられる。図35は、フィルタの前エッジが適用される各空間位置について、各フィルタリング動作が実行される時刻を示す数字を表わしている。

【0261】フィルタリングの順序はフィルタの前エッジで決まるので、フィルタリングは、あるツリーの係数の全部を生成しないうちに次のツリーの係数のどれかを生成する。ツリーの子のフィルタリングは親のフィルタリングの前に行なわれ、低域通過フィルタリングは対応の高域通過フィルタリングの前に行なわれる。フィルタリングは、Aグループの係数に作用する。

【0262】レベル1の水平フィルタリングは時刻0から時刻7の期間に実行され、その結果は一時バッファに格納される。(各空間位置から2個の係数が得られる。)時刻2から時刻9の期間に、(第2のフィルタに

51

より) 垂直フィルタリングが、バッファ内のデータ及びメモリから与えられる前水平フィルタリングによるデータに対して行なわれる(各空間位置毎に2回)。垂直フィルタリングは、2番目の水平フィルタリングが完了したら直ぐに開始できる。HH, HL, LH係数は(適当な時刻に) コンテキストモデルへ出力できる準備ができてい。LL係数は次のレベルで使われる。

【0263】フィルタが2個だけでは、レベル1の水平フィルタリングが完了して1つのフィルタが利用できるようになる時刻8まで、レベル0の水平フィルタリングは開始できない。レベル0の垂直フィルタリングが完了し必要データ全部が得られてから1サイクル後の時刻10まで、レベル0の水平フィルタリングは終了できない。次の時刻11と時刻12の間、レベル1の垂直フィルタリングを行なうことができる。

【0264】下記表1は、各時間単位における各フィルタの動作をまとめて示す。各エントリーの形式は、レベル番号、水平または垂直("H"または"V")、そして前エッジの空間位置である。垂直フィルタリング操作の入力は、添字"L"または"H"を用いて低域、高域が区別されている。なお、両方のフィルタは同じものであるから、一方のフィルタを水平フィルタリングに、他方のフィルタを垂直フィルタリングに、割り当てる必要はない。

【0265】

【表1】

表 1

Time	Filter 1	Filter 2
0	1H(0,0)	(idle)
1	1H(0,1)	
2	1H(2,0)	1V <sub>L</sub> (0,0)
3	1H(2,1)	1V <sub>H</sub> (0,0)
4	1H(0,2)	1V <sub>L</sub> (0,2)
5	1H(0,3)	1V <sub>H</sub> (2,0)
6	1H(2,2)	1V <sub>L</sub> (0,2)
7	1H(2,3)	1V <sub>H</sub> (0,2)
8	0H(0,0)	1V <sub>L</sub> (2,2)
9	(idle)	1V <sub>H</sub> (2,2)
10	0H(0,1)	(Idle)
11	(idle)	0V <sub>L</sub> (0,0)
12		0V <sub>H</sub> (0,0)
13		(idle)
14		
15		

52

【0266】レベル1水平フィルタリングは、次の入力画素群に対して時刻11に再び開始することができるけれども、そうするとフィルタを入出力レートより高速に動作させることになる。本発明では、そのようにしないで、フィルタは遊び(アイドル)になり、そして次の入力画素群は時刻16に始まることになる。遊びのフィルタリングサイクルはメモリ転送のために利用される。遊びサイクルは、必要ならば、各入力画素群に対するフィルタリングの終わりに発生させるのではなくて、フィルタリングサイクル中に分散させてもよい。

【0267】2レベルの場合を考慮に入れて3レベルの場合を表2に示す。情報を1ページに表わして読み易くするため、2または4時間単位の連鎖が使われている。

【0268】

【表2】

表 2

Time	Filter 1	Filter 2
0-3	2H(0,0), 2H(0,1), 2H(2,0) 2H(2,1)	(idle), (idle) 2V <sub>L</sub> (0,0), 2V <sub>H</sub> (0,0)
4-7	2H(4,0), 2H(4,1), 2H(6,0) 2H(6,1)	2V <sub>L</sub> (2,0), 2V <sub>H</sub> (2,0), 2V <sub>L</sub> (4,0), 2V <sub>H</sub> (4,0)
8-11	2H(0,2), 2H(0,3), 2H(2,2) 2H(2,3)	2V <sub>L</sub> (6,0), 2V <sub>H</sub> (6,0), 2V <sub>L</sub> (0,2), 2V <sub>H</sub> (0,2)
12-15	2H(4,2), 2H(4,3), 2H(6,2) 2H(6,3)	2V <sub>L</sub> (2,2), 2V <sub>H</sub> (2,2), 2V <sub>L</sub> (4,2), 2V <sub>H</sub> (4,2)
16-19	2H(0,4), 2H(0,5), 2H(2,4) 2H(2,5)	2V <sub>L</sub> (6,2), 2V <sub>H</sub> (6,2), 2V <sub>L</sub> (0,4), 2V <sub>H</sub> (0,4)
20-23	2H(4,4), 2H(4,5), 2H(6,4) 2H(6,5)	2V <sub>L</sub> (2,4), 2V <sub>H</sub> (2,4), 2V <sub>L</sub> (4,4), 2V <sub>H</sub> (4,4)
24-27	2H(0,6), 2H(0,7), 2H(2,6) 2H(2,7)	2V <sub>L</sub> (6,4), 2V <sub>H</sub> (6,4), 2V <sub>L</sub> (0,6), 2V <sub>H</sub> (0,6)
28-31	2H(4,6), 2H(4,7), 2H(6,6) 2H(6,7)	2V <sub>L</sub> (2,6), 2V <sub>H</sub> (2,6), 2V <sub>L</sub> (4,6), 2V <sub>H</sub> (4,6)
32-35	1H(0,0), 1H(0,1), 1H(2,0) 1H(2,1)	2V <sub>L</sub> (6,6), 2V <sub>H</sub> (6,6), 1V <sub>L</sub> (0,0), 1V <sub>H</sub> (0,0)
36-39	1H(0,2), 1H(0,3), 1H(2,2) 1H(2,3)	1V <sub>L</sub> (2,0), 1V <sub>H</sub> (2,0), 1V <sub>L</sub> (0,2), 1V <sub>H</sub> (0,2)
40-43	0H(0,0), (idle), 0H(0,1), (idle)	1V <sub>L</sub> (2,2), 1V <sub>H</sub> (2,2), (idle), 0V <sub>L</sub> (0,0),
44-47	(idle)	2V <sub>H</sub> (0,0), (idle), (idle), (idle)
48-51		(idle)
52-55		
56-59		
60-63		

【0269】表3は4レベルの場合を示す。1グループの係数につき256時間ユニットあるので、単純にするためフィルタリングのレベルと方向だけが表示されてい

る。

【0270】

【表3】

表 3

Time	Filter 1	Filter 2
0-1	Level 3 Horizontal	(idle)
2-127	Level 3 Horizontal	Level 3 Vertical
128-129	Level 2 Horizontal	Level 3 Vertical
130-159	Level 2 Horizontal	Level 2 Vertical
160-161	Level 1 Horizontal	Level 2 Vertical
162-167	Level 1 Horizontal	Level 1 Vertical
168	Level 0 Horizontal	Level 1 Vertical
169	(idle)	Level 1 Vertical
170	Level 0 Horizontal	(idle)
171	(idle)	Level 0 Vertical
172	(idle)	Level 0 Vertical
173-255	(idle)	(idle)

【0271】本発明のフィルタリング及びメモリサブシステムの出力は係数の系列であり、これは本発明のビット・シグニフィカンス埋め込み符号化を受ける。

【0272】<B. 1パスシステム用コンテキストモデル>本発明の一実施例である1パスシステム用ビット・シグニフィカンス埋め込みコンテキストモデルでは、各ツリーが4つの部分に分けて処理される。ツリーのルートである最高レベルのLH係数は、1パス水平順序符号化によって符号化される。ルートの3個の子から始まる3個のサブツリー(subtree)である最高レベルのH

H, HL, LH係数は、1パス統合空間/周波数モデリング及び1パス周波数ベースモデリングの両方によって処理される。ビット・シグニフィカンス埋め込みコンテキストモデルがデータ全部を処理するより前に符号化データが出力されるように、係数は符号化される。

【0273】<1パス・シグニフィカンス(significance) ツリー>ゼロツリーコンテキストモデルは1パスシステムに利用できない。ゼロツリーは全ての係数を保持する1つのリスト(または複数のリスト)を必要とし、またゼロツリーはリストに複数のパスを作る。もう一つ

55

の周波数ベースモデルである1パス・シグニフィカンスツリーは、全係数を保持するリストを全く必要としない。1パス・シグニフィカンスツリーとゼロツリーとの間のもう一つの相違点は、シグニフィカンスツリーは全ての子を処理した後に、その親を処理しつつデシジョンを生成するのに対し、ゼロツリーは親を先に処理することである。本発明のコンテキストモデルが図36にブロック図の形で示されている。このコンテキストモデル1700は、符号／絶対値ユニット109(図37)とシグニフィカンス(significance)ユニット1702という2つの処理ユニットを含む。コンテキストモデル1700はまた、絶対値メモリ1701とツリー(tree)メモリ1703という2つのメモリを(メモリ制御論理も)用いる。これら2つのメモリユニットのそれぞれは、複数の記憶エリアによって高速動作中に交互に使用できるように構成してもよい(すなわち、一方がデータ書き込み中である時に、他方は読み出し中であるかまたは空である)。

【0274】絶対値メモリ1701は、ツリー中の係数をシグニフィカンスに基づいた順序に、例えば、係数の絶対値に基づいた順に並べ替える。この並べ替えは、各可能な絶対値のためのキュー(queue)を作ることによって達成される。シグニフィカンスユニット1702は、係数をシグニフィカンス(例えば絶対値)の順に受け取り、A-パスアルゴリズムを処理する符号化器のためのデシジョンを生成する。ツリーメモリ1703はシグニフィカンスユニット1702に接続され、オールゼロの後のゼロツリーを削除する。

【0275】以下の説明は、係数が18ビットであり、かつ入力データが4レベル分割されていると仮定している。

【0276】符号-絶対値フォーマッティングユニット109の一実施例が図37に示されており、これは入力

56

した係数を符号-絶対値形式へ変換する。符号-絶対値フォーマッティングユニット109は、18ビットの係数を受け取るように接続されており、インバータ1801、マルチプレクサ(MUX)1802、プライオリティ(priority)エンコーダ1803、及びカウンタ1804からなる。符号-絶対値ユニット109は、シグニフィカンス(significance)表示(例えば5ビット値)、入力係数の仮数(例えば17ビット)、入力係数の符号(sign)(例えば1ビット)、及びカウンタ1804からのインデックス(例えば7ビット)を出力する。

【0277】MUX1802は、符号-絶対値フォーマッティングユニット109に直接入力した係数の17ビットと、この17ビットをインバータ(2の補数器)1801により反転したものを受け取るように接続されている。MUX1802の選択入力に入る符号ビット(係数のビット17)に基づき、2つの入力のうちの正のものが仮数として出力される。

【0278】符号-絶対値フォーマッティングユニット109はプライオリティエンコーダ1803を使い、各係数の最初の有効ビットを決定する。各係数の最初の有効ビットに基づいて、1つのシグニフィカンスレベルを係数に関係付けることができる。

【0279】カウンタ1804は、1つのインデックスを現在のツリー要素に関係付けするために利用される。4レベル分割の場合、そのインデックスは0から84まで変化する(1つのサブツリーの要素数は $1+4+16+64=85$ であるから)。入力係数はツリー順であり、この例では、最初に親、最後が子の順序であると仮定している。異なった分割レベルによる係数は、表4に示す通りである。

【0280】

【表4】

表 4

Level	Coefficients index
0	0
1	1, 22, 43, 64
2	2, 7, 12, 17, 23, 28, 33, 38, 44, 49, 54, 59, 65, 70, 75, 80
3	3...6, 8...11, 13...16, 18...21, 24...27, 29...32, 34...37, 39...42, 45...48, 50...53, 55...58, 60...63, 66...69, 71...74, 76...79, 81...84

【0281】図38は、絶対値メモリ1701の一実施例のブロック図である。1つのカウンタ(1900~1916)とメモリ(1920~1936)が、可能な各シグニフィカンスレベルに関係付けられている(ただし、符号化される必要のないゼロ係数に対しては何も必要でない)。例えば、カウンタ1916とメモリ1936はシグニフィカンス(significance)レベル17に関係付けられている。一実施例では16のシグニフィカン

スレベルがある。したがって、17個のカウンタ1900~1916と、それに関連した17個のメモリ1920~1936がある。

【0282】一実施例では、各メモリは、サブツリーの可能な各係数につき、85のロケーションを持たなければならない(各サブツリーは85個の係数を含むから)、しかし、そのメモリサイズは便宜上、2のべき乗、例えば128に丸められるであろう。メモリの各エ



57

ントリーは1ビットの符号ビット、7ビットのインデックスビット及びNビットの絶対値ビットを格納できる（Nはシグニフィカンスレベルである）。固定幅のメモリを使いたいときには、シグニフィカンスレベル16と0のエントリー、シグニフィカンスレベル15と1のエントリー、等々を結合することができ、そうすることにより、各ワードは合計32ビットの2つのエントリーを含む。当然のことながら、シグニフィカンスレベル数が奇数の場合、あるワードは1つのエントリー（この例ではレベル7）しか含まない。

【0283】符号—絶対値フォーマッティングユニット109より受け取った符号（sign）、インデックス及び仮数の値は、適切なメモリの、それに関係付けられたメモリカウンタにより与えられるアドレスに書き込まれる。この関係付けられたカウンタは、該当シグニフィカンスレベルの次の係数が次ロケーションに格納されるようインクリメントされる。

【0284】記憶内容は、シグニフィカンスの降順にメモリ1902～1926のそれぞれより読み出される。各係数の出力は、その仮数、符号（sign）及びインデックス出力からなる。最高のシグニフィカンスレベル（例えばレベル16）のカウンタが0でなければ、このカウンタはデクリメントされ、メモリはそのアドレスが読み出される。これはカウンタ値が0になるまで繰り返される。そして、次のシグニフィカンスレベル（例えばレベル15）が検討される。全てのカウンタが0までデクリメントされ全部のメモリが空になるまで、各シグニフィカンスレベルが順に検討される。

【0285】リアルタイムのシステムにおいては、2バンクのカウンタ及びメモリを用いるのが望ましいかもしれない。そうすれば、一方のバンクを、他方のバンクが出力のために使われている時に入力のために用いることができる。

【0286】カウンタは、LIFO（後入れ先出し）が実現されるように関連したメモリをアドレスする。LIFOは、サブツリーが親から先に入力される場合には適する順序である。そうではなく、サブツリーが子から先に入力されるのであれば、カウンタの動作はFIFO（先入れ先出し）を実現するように変更されることになる。

【0287】図39はシグニフィカンスユニット1702の一実施例のブロック図である。図39において、インデックスカウンタ2001はサブツリーの各係数を、子から先に順に進めるために用いられる。一実施例では、インデックスカウンタ2001は84に初期化されてから0までカウントダウンする。シグニフィカンス（significance）カウンタ2004は、最大のシグニフィカンスレベル（この例では16）から始まって、インデックスカウンタが1サイクルする（84に戻る）毎にカウントダウンすることにより、ビットプレーンを管理

58

する。ある特定のインデックスのレベルが、前記表4に示した機能を遂行するインデックス—レベル論理2003によって決定される。

【0288】絶対値メモリ1701は、シグニフィカンスカウンタ2004の出力によりイネーブルされたメモリ内にある次の係数のインデックス、絶対値及び符号を提供する。メモリから入力したインデックスがインデックスカウンタ2001のインデックス出力と同一であると、一致論理2002が非ゼロ出力表示をアサートする。この非ゼロ出力表示は、次サイクルで絶対値メモリが次のインデックス等を供給すべきことを意味する。一致がとれないときには、不一致表示がデシジョン（decision）ジェネレータ2008へ送られる。

【0289】一実施例では、フラグ0、フラグ1、フラグ2として示した3個のフリップフロップ2005、2006、2007が、非ゼロデータの管理のために用いられ、また、これらフリップフロップは分割レベル0、1、2にそれぞれ割り当てられている。なお、必要なフリップフロップの個数は、分割レベル数より1つ少ない。フリップフロップ2005～2007は、最初にクリアされる。一致論理2002から出る非ゼロ信号がアサートされた時に、フリップフロップ2005～2007中の現在のレベルより小さいレベルに割り当てられたフリップフロップがセットされる。現在のレベルに割り当てられたフリップフロップはクリアされる。レベルはインデックス—レベル論理2003によって与えられるが、このインデックス—レベル論理2003はインデックスカウンタ2001によって与えられたインデックスに応じてレベルを提供する。

【0290】”符号化済み”フラグ（ある実施例ではレジスタファイル）が各インデックス毎に1ビットずつ記憶される。非ゼロ信号がアサートされた時に、符号化済みフラグ記憶内の現在インデックスカウンタ値に関連したビットがセットされる。そうでなくて、シグニフィカンスカウンタ値が最大値になれば、関連したビットはクリアされる。そうでなければ、そのビットの値はそのままの値を維持する。符号化済みフラグ記憶から出る符号化済み出力信号は、現在のインデックスに関連したビットの新値と同じである。なお、他の実施例では、符号化済みフラグは用いられず、したがって符号化済み信号も全く用いられない。

【0291】一実施例では、デシジョンジェネレータ2008は、現在レベルが3であり、かつ前レベルが3でなかったか判定する。この判定に応じて、デシジョンジェネレータ2008は開始出力をアサートし、開始レベル出力は前レベルである。非ゼロ信号がアサートされたならば、デシジョンジェネレータ2008は”有意”なるデシジョンを出力し、また、符号（00、01）と仮数を出力する。そうでなくて、符号化済み入力がアサートされたならば、デシジョンは出力されない。そうでな

59

くて、現在レベルに割り当てられたフラグフリップフロップがセットされたならば、デシジョンジェネレータ2008は”無意、有意な子あり”なるデシジョン(10)を出力する。そうでなければ、デシジョンジェネレータ2008は、”無意、かつ子も無意”なるデシジョン(11)を出力し、かつオールゼロ信号をアサートする。

【0292】なお、周波数ベースモデリングと水平1パス統合空間一周波数モデリングの両方を実現するためには、シグニフィカンスユニット1702に対し以下の変更がなされる。シグニフィカンスカウンタ2004はある閾値と比較され、カウンタ値が同閾値を超えるときにのみオールゼロ信号がアサートされる。

【0293】一実施例では、ツリーメモリ1703(図40、後述)に対するシグニフィカンスカテゴリー入力、シグニフィカンスカウンタ2004の出力である。この実施例のコンテキストモデル(すなわちビット・シグニフィカンス埋め込みユニット)では、シグニフィカンスカテゴリーはビットプレーン数に基づいており、17の異なったシグニフィカンスカテゴリーがある。これは任意に選んでよい。別の実施例では、シグニフィカンスカテゴリーを減らすためビットプレーンが結合されるかもしれない。また、より多くのシグニフィカンスカテゴリーを作るために、レベル情報をビットプレーン情報に追加することもできる。シグニフィカンスカテゴリーを増やすと、より優れた損失性圧縮が可能になるであろう。他方、シグニフィカンスカテゴリーが少ないほうが、ハードウェアの複雑さが軽減されるであろう。

【0294】図40は本発明のツリーメモリユニットの一実施例のブロック図である。図40において、メモリ2101は、デシジョンと、可能な各デシジョンのためのシグニフィカンス表示を記憶するための適切なスペースを有する。一実施例では、4レベル分割、7シグニフィカンスレベルの場合で、メモリ2101のロケーション数は $85 \times 17 = 1455$ である。

【0295】メモリ2101をアクセスするためアドレスが生成される。カウンタ2102は最初は0である。デシジョンジェネレータ2008がオールゼロ入力をアサートしない時に、カウンタ2102の値がメモリのアドレスのために用いられる。デシジョンジェネレータ2008が開始入力をアサートすると、カウンタ2102の現在値は開始レベルに応じてレジスタ2110~2112中の1つのレジスタに格納される。開始レベルは一種の選択機構として作用する。そして、カウンタ2102はインクリメントされる。

【0296】デシジョンジェネレータ2008がオールゼロ入力をアサートした時に、レベル入力により選択されたレジスタ(すなわち2110、2111、2112)中の値がメモリ2101のアドレスのために用いられ、この値に1を加えた値がカウンタ2102にロード

60

される。これは、無意の親の無意の子のために使用されたメモリロケーションを無視させる。

【0297】メモリ出力期間に、出力すべきロケーションのアドレスを提供するためカウンタ2102はデクリメントされる。出力(及びデクリメント)はカウンタ2102が0になった時に止まる。メモリ2101の出力はエントロピー符号化器に受け取られ、エントロピー符号化器は指定されたシグニフィカンスでデシジョンを適切に符号化する。

【0298】リアルタイム動作のためには、2個のツリーメモリユニットを用い、一方を入力に利用し、同時に他方を出力に利用するようにすることができる。

【0299】<係数アラインメント(alignment)>本発明の一実施例においては、ゼロツリーコンテキストモデルは非正規化 $1 + Z^{-1}$ 低域通過フィルタを使用する。しかし、ゼロツリーコンテキストモデルは、例えば次に示したような正規化フィルタと一緒に使用し得る。

【0300】

【数36】

$$\frac{1 + Z^{-1}}{\sqrt{2}}$$

【0301】正規化フィルタを使用するためには、図41のフォワード・ウェーブレットフィルタ1000と(周波数ベース)コンテキストモデル105の間のアラインメント(alignment)ユニット2200のようなアラインメントユニットを用いて、非正規化フィルタにより獲得したエネルギー(あるいは失ったエネルギー)を補償し、圧縮を改善することができる。アラインメントは損失性動作のための非一樣量子化を許容するので、アラインメントは損失性画像復元の視覚的品質を改善できる。一次元の場合、ツリーの各レベルからの係数のアラインメントは様々であろう(除数= $2^{1/2}$ , 2,  $2 \times 2^{1/2}$ , 4, 乗数= $2 \times 2^{1/2}$ , 2,  $2^{1/2}$ , 1)。二次元の場合、除数は2, 3, 8, 16、乗数は8, 4, 2, 1であろう。

【0302】アラインメントは単に同様の符号化用バイナリデシジョンをグループにするためであるので、厳密な正規化値を用いることは重要でない。復号期間にはアラインメントを逆にしなければならないので、乗算と除算の両方が必要になる。2のべき乗の乗数(因数)/除数を用いると、ハードウェアによる効率的なシフト操作を行なえるようになる。係数に2のべき乗が掛け合わされる場合、シグニフィカンスの小さい追加された0ビットは符号化する必要がない。

【0303】しかし、アラインメント乗数/除数を2のべき乗に制限する代わりに、 $2^{1/2} \approx 1.5$ または $2^{1/2} \approx 2 \div 1.5$ というような近似を、以下の方法とともに用いることができる。乗数/除数で係数を乗算/除算するのではなく、代わりに”重要な”係数だけが乗

61

数/除数によりスケール (scale) されてよい。符号—絶対値フォーマットユニットは、図42に示すように、(1) 最も上位の"1"ビットの次位ビットも"1"のときに最も上位の"1"ビットの位置を返し、(2) そうでなければ最も上位の"1"ビットの一つ下の位置を返す"1.5"プライオリティエンコーダ2301を含むように変更し得る。3ビット入力の場合の"1.5"プライオリティエンコーダの真理値表を表5に示す。

【0304】

【表5】

表 5

Input (Binary)	Output
001	0
010	0
011	1
100	1
101	1
110	2
111	2

【0305】現在のインデックス値で指示される係数のレベルに応じて、マルチプレクサ2302は標準的なプライオリティエンコーダ1803または"1.5"プライオリティエンコーダ2301のどちらかより与えられるシグニフィカンス (significance) を選択する。"

1.5"アラインメントが用いられる時には常に、仮数はN+1ビットである (Nはシグニフィカンス値)。そうでない時には、仮数はNビットである。

【0306】アラインメントユニット2200は、シフタとして作用する2入力乗算器からなり、1または2によるアラインメントを実施できる。これと、符号—絶対値フォーマットユニットにより提供される"1.5"アラインメントとを組み合わせることで、1, 1.5, または3のアラインメントが可能である。1, 1.5または3は、単純であるので (すなわち2のべき乗である)、一次元信号のために必要な乗数のよい近似値である。(画像のような二次元信号の場合、それら数はより単純である。) 復号期間において、"1.5"プライオリティエンコーダが用いられる時には仮数の第(N+2)ビット (符号されない) は第(N+1)ビットの補数である。

【0307】係数アラインメントを、ゼロツリーのチューニング及びより精度の高い非一様量子化のために利用できる。画像 (二次元信号) の場合、RTS変換の一実施例は、図56に示した数を周波数帯域に掛け合わせるによって、係数を調整する。これらの数を掛けると、RTS変換が、その正確な復元ウェーブレットの極

62

めて精度のよい近似となる。

【0308】エントロピー符号化器は、アラインメントプロセスが効率的になるよう考慮しなければならない。

【0309】部分ビットプレーンによる周波数ベースコンテキストモデル

周波数ベースモデリングの別の方法は、部分ビットプレーンもしくは部分的な有意ビット (bits of significance) を利用する。これを実現する一つの方法は、各ビットプレーンを2回処理することであり、その結果、パスとしてA1-パス、B1-パス、A0-パス、B0-パスがある。なお、これらパスの名称は、A1-パスが"11"で始まる係数を扱い、A0-パスが"10"で始まる係数を扱うことから選ばれた。

【0310】ビットプレーンSに対するA1-パスの期間において、A-グループの係数が有意であるのはビットS, S-1がともに非ゼロのときだけである。A2-パスの期間において、A-グループの係数はビットSが非ゼロのときに有意である。最上位の2ビットは知れているので、B1-パスとB0-パスはS-1ビットを処理するだけでよい (S=0は最下位ビットプレーンであると仮定する)。

【0311】1つおきの部分ビットプレーンは、因数が1.5または2/1.5と異なるから、異なるレベルに対するアラインメントは、各レベル毎に必要な部分ビットプレーンをグループにすることによって達成できる。

【0312】部分ビットプレーンは、周波数ベースコンテキストモデルに利用された親/子関係によってデータのより精密なモデリングをもたらす。さらに精密なモデリングのために、3つ以上のパス、例えば4つまたは8つのパスを用いてもよい。例えば、4パスの場合、A1-1-パスは"111"で始まる係数を扱うことになる。そして他のパスは"110", "101", "100"を扱うことになる。より精度の悪いモデリングもまた利用されるかもしれない。例えば、1つおきのビットプレーンに対しのみ1つのパスが行なわれるかもしれない。このような低精度モデリングの場合には、より多くのビットがB-グループによって符号化される。

【0313】<C. 1パスシステムのための符号化器及びメモリ/チャネル管理>データ全部をメモリに格納するシステム及びデータをチャネルで伝送するシステムに関して、1パスシステムにおける符号化データのメモリ管理を提案する。1パスシステムでは、符号化データは"埋め込みカジュアル (embedded casual)" アクセスすることができて、シグニフィカンスの高いデータを損なわずにシグニフィカンスの低いデータを廃棄できるよう格納されなければならない。符号化データは可変長であるので、動的メモリ割り当てを利用できる。

【0314】本発明の一実施例では、埋め込み符号化スキームは18個のビットプレーンを用い、したがって18個の重シグニフィカンスレベルをデータに割り当て

る。1パスシステムの符号化器は、“埋め込みカジュアル (embedded casual)” でなければならない。すなわち、あるビットプレーンに対応するイベントの復号に、それにより下位のビットプレーンの情報を必要としない。1パスの場合、普通、あるツリーのビット全部が符号化された後に次のツリーのビットが符号化されるから、シグニフィカントの異なるビットが分離されない。内部状態 (internal state) を用いるハフマン符号化器のような符号化器にとって、このことは問題ではない。しかし、多くの圧縮率の優れた高度な圧縮器は内部状態を用いる。

【0315】これら符号化器に関する当該問題を解決する一方法は、18個の異なる符号化器、多分Q-コーダチップ、を用いることである。9個のQ-コーダチップを使用できるであろう手法が、“Data Compression for Recording on a Record Medium”なる発明の名称で1992年3月17日発行された米国特許第5,097,261号 (Langdon, Jr) に述べられている。より優れた方法は、単一の物理符号化器で様々な仮想符号化器を実現するためにパイプライン方式の符号器、例えば“Method and Apparatus for Parallel Decoding and Encoding of Data”なる発明の名称で1993年1月10日受理された米国特許出願第08/016,035号に述べられている符号化器を用いる。このような符号化器では、各確率毎の複数のビットジェネレータ状態がデータの一部に割り当てられる。例えば、18ビットデータの場合、18個の状態中の各状態がある1つの特定のビットプレーンに割り当てられることになろう。符号化器内部のレジスタもまたデータの各部分に割り当てられる。この符号化器においては、インターリービングは行なわれない。すなわち、データの各部分は単にビット詰めされる。

【0316】複数の物理符号化器または仮想符号化器を備えた実施例では、データの各部分にメモリが割り当てられる。圧縮が完了した時に、割り当てられたメモリとその内容を記述する連結リストが結果として得られる。

【0317】メモリがオーバーフローすると、メモリ割り当てルーチンはより重要な情報をそれより重要でないデータに上書きさせる。例えば、数値データの最下位ビットが初めに上書きされることになろう。メモリの割り当てられ方を記述する情報が、符号化データのほかに、格納されなければならない。

【0318】図43は、3つのシグニフィカンスカテゴリーのための動的メモリ割り当てユニットの例を示す。本発明をいたずらに難解にしないため3カテゴリーしか述べないが、一般的には、8カテゴリー、16カテゴリー、18カテゴリーというように、もっと大きな数のカテゴリーが用いられるであろう。レジスタファイル (または他の記憶手段) 481が、各シグニフィカンスカテゴリー毎のポインタ (現在ポインタ) のほかに、次の空きメモリロケーションを指示するための別のポインタ

(フリーポインタ) を保持する。メモリ482は一定サイズのページに分割される。

【0319】最初、1つのシグニフィカンスカテゴリーに割り当てられた各ポインタは、あるメモリページの先頭を指し示し、また、フリーポインタは次に利用できるメモリページを指し示す。シグニフィカンスカテゴリーで区別された符号化データは、対応したポインタによりアドレスされたメモリロケーションに格納される。そして、このポインタは、次のメモリロケーションを指すようにインクリメントされる。

【0320】ポインタが現在ページの最大値に達した時に、フリーポインタに格納されている次の空きページの先頭のアドレスが、リンクとして現在ページと一緒に格納される。一実施例では、この目的のために、符号化データのメモリまたは独立したメモリもしくはレジスタファイルが用いられるかもしれない。次に、現在ポインタは次のフリーページを指すように設定される。フリーポインタはインクリメントされる。これらのステップにより、1つの新しいメモリページがある特定のシグニフィカンスカテゴリーに割り当てられ、かつ、ある共通のシグニフィカンスカテゴリーのためのデータを含むメモリページのリンクを得られる結果、復号期間に割り当て順序を確認することができる。

【0321】メモリの全ページが使用中であり、しかも、メモリ内のシグニフィカンスが最小のデータよりシグニフィカンスが大きいデータがさらに存在する時には、メモリの再割り当てが行なわれるかもしれない。そのような再割り当てのための3つの手法について述べる。いずれの手法にあっても、最小のシグニフィカンスのデータに割り当てられているメモリが、それより高いシグニフィカンスのデータへ再割り当てされ、もはや最小シグニフィカンスのデータは格納されない。

【0322】第1の手法は、最小シグニフィカンスのデータに現在使われているページが単純にシグニフィカンスの高いデータに割り当てられるだけである。最も一般的なエントロピー符号化器は内部状態情報を利用するので、そのページに前に格納されていた最小シグニフィカンスのデータは全部失われる。

【0323】第2の手法は、最小シグニフィカンスのデータによって現在使われているページは、それより大きいシグニフィカンスのデータに割り当てられる。第1の手法と違い、ポインタは当該ページの末尾を指すように設定され、大きいシグニフィカンスのデータが当該ページに書き込まれるにしたがい対応ポインタはデクリメントされる。この手法は、大きなシグニフィカンスのデータがページ全体を必要としないときには、ページの先頭にある最小シグニフィカンスのデータが保存されるという利点を有する。

【0324】第3の手法は、最小シグニフィカンスのデータの現在ページを再割り当てするのではなく、最小シ

65

グニフィカンスのデータの任意のページを再割り当てすることができる。そのためには、全ページの符号化データが独立に符号化される必要があり、このことは圧縮率の低下を招くかもしれない。また、全てのページの前頭に対応する非符号化データが識別される必要もある。最小グニフィカンスのデータの任意のページを捨てることのできる、より大きな量子化のフレキシビリティを得られる。

【0325】画像の複数領域にわたって一定の圧縮率を達成するシステムでは、上記第3の手法は特に魅力的であろう。ある指定した数のメモリページを画像の1つの領域に割り当てることができる。グニフィカンスの小さなデータが保存されるか否かは、特定の領域において達成される圧縮率によって決まる。なお、ある領域に割り当てられたメモリは、非損失性圧縮が必要とするメモリがそのメモリ量より少ないならば、完全には利用されない。画像のある領域に対し一定の圧縮率を達成すれば、その画像領域へのランダムアクセスをサポートすることができる。

【0326】圧縮が完了した時に、そのデータを必要ならばグニフィカンス順にチャンネルまたは記憶装置へ送ってよい。そうすれば、様々なリンク(link)とポインタはもう必要ではなく、また、マルチパス復号を行ってもよい。あるいは、1パス復号のために、各グニフィカンス毎にデータへのポインタを保存することができる。

【0327】用途によっては、いくつかのグニフィカンスカテゴリは使われないかもしれない。例えば、12ビットの医用画像に対して16ビットの圧縮器が使用されるかもしれないが、その場合、ビットプレーン15...12に対応したグニフィカンスカテゴリは使用されないであろう。ページが大きく、かつ、多くのグニフィカンスカテゴリが使用されない例では、(いくつかのカテゴリが使われないことをシステムが予め知らない時には)それらの使用しないカテゴリにメモリを割り当てる必要はないのであるから、メモリを浪費することになる。このメモリ浪費に対するもう一つの解決策は、各グニフィカンスカテゴリ毎のカウントを保持するための小メモリ(もしくはレジスタ)を使用することであろう。このカウントは、他のデシジョンが発生する前に発生した"無意、有意な子なし"デシジョンの数を記録するであろう。これらのカウンタの格納のため必要となるメモリは、不使用のグニフィカンスカテゴリに利用されるメモリと"相殺"されなければならない。

【0328】システムにおいて利用可能なメモリ全量をより上手に利用するため、各ページにその両端からデータを書き込む機能を利用することができる。全ページが割り当てられている時に、一端側に十分な空きスペースのあるページを、その端から利用するように割り当てる

66

ことができる。ページの両端を利用する機能は、2種類のデータがぶつかるロケーションの管理コストと比較衡量されなければならない。もっとも、一方のデータ種類が重要でなく単純に上書きされてよい場合は別である。

【0329】<チャンネルの利用>データがメモリに格納される代わりにチャンネルで伝送され、かつ、固定サイズのメモリページが利用される(1つのグニフィカンスカテゴリにつき1ページだけ必要とされる)システムにおいて、あるメモリページが一杯になった時に、そのページはチャンネルで伝送され、そして、伝送されるとすぐにメモリロケーションは再び利用できる。用途によっては、メモリのページサイズはチャンネルで用いられるデータパケットのサイズまたはパケットサイズの倍数とすることができる。(なお、一実施例では、1グニフィカンスレベルあたり2ページを使用できるので、一方のページにデータを書き込みながら他方のページをチャンネルへ出力するため読み出すことができる。)

ある通信方式では、例えばATM(非同期転送モード)では、パケットにプライオリティを割り当てることができる。ATMは、プライマリとセカンダリの2つのプライオリティレベルを有する。セカンダリパケットは十分な帯域幅を利用できるときだけ伝送される。どのグニフィカンスカテゴリがプライマリであるか、どのグニフィカンスカテゴリがセカンダリであるかを判断するために、閾値を利用することができる。もう一つの方法は、ある閾値よりグニフィカンスの小さいグニフィカンスカテゴリを伝送しないように、符号化器において閾値を利用することである。

【0330】<ピーク誤差を制限した損失性圧縮>ある種の用途では、完全な(非損失の)復元は必要でない。指定された最大のピーク誤差で圧縮を成し遂げることが望ましいかもしれない。ピーク誤差を $\pm E$ と仮定する。これは、圧縮データを切り捨て、所望の精度を得るのに必要でないグニフィカンスの小さなデータ全てを廃棄することによって達成できる。

【0331】指定された最大ピーク誤差の圧縮を遂行するもう一つの方法は、 $2 \times E + 1$ 以下の値で、圧縮すべき画像の各画素を除算(整数除算)することである。復元中に、画像の各画素は

$$\text{出力画素} = (2 \times E + 1) \times \text{入力画素} + E$$

と処理される。(あるいは、伸長中に $E$ を加算する代わりに、圧縮中に $2 \times E + 1$ で割算する前に減算が行なってもよい。)

指定された最大ピーク誤差の圧縮を達成するもう一つの方法は、除算と乗算をシフトで置き換えることである。

そのシフト量は、

【0332】

【数37】

$$\lceil \log_2(2xE+1) \rceil$$

67

【0333】である。シフト操作は便利であるので、(ピーク誤差に代わる)より好ましい誤差仕様は $(-2^{n-1} \leq \text{エラー} \leq 2^n)$ なる形の誤差であろう。

【0334】上に述べたことは、損失性画像圧縮の技術分野において周知である係数の量子化と混同すべきでない。多くの損失性圧縮システム(例えばJPEG)においては、変換領域係数は最大ピーク誤差を割り付けられるが、これは画像のピーク誤差を間接的に規制するに過ぎない。決定的な違いは、本発明が画素に対して量子化を行ない、かつ、係数の非損失性圧縮を使うことである。

【0335】変換領域量子化も利用することができる。多くの係数は、変換の複数レベルにわたるピーク誤差の伝搬に対して影響がある。子のない高域通過係数に関するピーク誤差への影響を決定することは容易である。

【0336】一次元信号を最大ピーク誤差 $\pm E$ で符号化することを考える。これは、最も精細な高域通過係数を $\pm 2E$ に量子化することで達成できる。二次元信号の対しては、高域通過フィルタが2回かけられるので、最も精細なHH係数を $\pm 4E$ に量子化すればよい。

【0337】入力画像の量子化による方法の一つの代案は、エントロピー符号化器に対するデシジョンを規制することである。一例は次のとおりである。各係数について、係数を0に設定しても、当該係数に影響されるいかなる画素にも最大誤差を超える誤差が生じないならば、当該係数は0に設定される。ある具体例では、特定の係数のみ、多分、子を持たないAC係数だけがテストされる。係数は、一度に一つしか検討しない贅沢なやり方で検討される。ほかのやり方は、係数の小グループを考え、グループの可能な最大の部分集合を0にするようにする。

【0338】

【表6】

表6 マスク

Bitplane	Mask (binary)
7	00000000
6	10000000
5	11000000
4	11100000
3	11110000
2	11111000
1	11111100
0	11111110

【0339】

68

【発明の効果】以上、実施例に関連して詳細に述べたところから明らかなように、本発明によれば、良好なエネルギー集中を得られる変換を用いた効率的な符号化装置と、それに関連した効率的な圧縮を可能にするウェーブレット変換フィルタとを実現することができ、したがって、効率的な非損失性/損失性データ圧縮/伸長システムを提供できる等の効果を得られる。

【図面の簡単な説明】

【図1】本発明の符号化システムの符号化部の一実施例を示すブロック図である。

【図2】本発明のビット・シグニフィカンス埋め込みブロックの一実施例を示すブロック図である。

【図3】ウェーブレット分析/合成システムのブロック図である。

【図4】非オーバーラップ最小長可逆フィルタを用いてフィルタリングする変換システムのフォワード変換とリバース変換の説明図である。

【図5】レベル1分割の結果の説明図である。

【図6】レベル2分割の結果の説明図である。

【図7】レベル3分割の結果の説明図である。

【図8】レベル4分割の結果の説明図である。

【図9】3レベル・ピラミッド変換の一例を示すブロック図である。

【図10】2次元・2レベル変換の一例を示すブロック図である。

【図11】2次元・2レベル変換の別の例を示すブロック図である。

【図12】本発明の可逆ウェーブレットを用いる圧縮・伸長システムを示すブロック図である。

【図13】本発明の可逆ウェーブレットを用いる強調及び分析システムを示すブロック図である。

【図14】ウェーブレット係数のツリー構造を示す図である。

【図15】本発明の単一リスト・ゼロツリー符号化プロセスの一実施例を示すフローチャートである。

【図16】図15に示したフローチャートの続きを示すフローチャートである。

【図17】本発明の単一リスト・ゼロツリー符号化プロセスの他の実施例を示すフローチャートである。

【図18】図17に示したフローチャートの続きを示すフローチャートである。

【図19】本発明のゼロツリー水平復号プロセスのためのA-パスの一実施例を示すフローチャートである。

【図20】本発明のゼロツリー水平復号プロセスのためのA-パスの他の実施例を示すフローチャートである。

【図21】本発明の単一リスト水平符号化プロセスのためのA-パスの一実施例を示すフローチャートである。

【図22】本発明の単一リスト水平符号化プロセスのためのA-パスの他の実施例を示すフローチャートである。

50

69

【図23】本発明の単一リスト水平復号プロセスのためのA-パスの一実施例を示すフローチャートである。

【図24】本発明の単一リスト水平復号プロセスのためのA-パスの他の実施例を示すフローチャートである。

【図25】本発明のゼロツリー水平符号化及び単一リスト水平符号化のプロセスのためのB-パスの一実施例を示すフローチャートである。

【図26】本発明のゼロツリー水平符号化及び単一リスト水平符号化のプロセスのためのB-パスの他の実施例を示すフローチャートである。

【図27】本発明のゼロツリー水平復号及び単一リスト水平復号のプロセスのためのB-パスの一実施例を示すフローチャートである。

【図28】本発明のゼロツリー水平復号及び単一リスト水平復号のプロセスのためのB-パスの他の実施例を示すフローチャートである。

【図29】本発明のフォワード・ウェーブレットフィルタの一実施例を示すブロック図である。

【図30】本発明のリバース・ウェーブレットフィルタの一実施例を示すブロック図である。

【図31】4レベル・ピラミッド分割のためのラインバッファ内の画像及び係数を説明するための図である。

【図32】フィルタ制御ユニットを使用する本発明のウェーブレットフィルタユニットの一実施例を示すブロック図である。

【図33】フィルタ制御ユニットを使用する本発明のウェーブレットフィルタユニットの他の実施例を示すブロック図である。

【図34】水平、垂直アクセスをサポートするためのメモリバンクの割り当てを2バンクの場合と4バンクの場合について示す図である。

【図35】2レベル分割の場合の1パスフィルタ動作を説明する図である。

【図36】本発明のコンテキストモデルの一実施例を示すブロック図である。

【図37】本発明の符号-絶対値フォーマットユニットの一実施例を示すブロック図である。

【図38】本発明の絶対値メモリユニットの一実施例を示すブロック図である。

【図39】本発明のシグニフィカンスユニットの一実施例を示すブロック図である。

【図40】本発明のツリーメモリユニットの一実施例を示すブロック図である。

【図41】ウェーブレット係数のアラインメントを説明するためのブロック図である。

【図42】"1.5"アラインメントを使用した本発明のシグニフィカンスユニットの一実施例を示すブロック図である。

【図43】1パス動作のための符号化データメモリの動的割り当ての例を示す図である。

70

【図44】本発明の符号化プロセスの一実施例を示すフローチャートである。

【図45】図44に示したフローチャートの続きを示すフローチャートである。

【図46】本発明の復号プロセスの一実施例を示すフローチャートである。

【図47】図46に示したフローチャートの続きを示すフローチャートである。

【図48】本発明の符号化プロセス及び復号プロセスのための各係数のモデリングプロセスの一実施例を示すフローチャートである。

【図49】図48に示したフローチャートの続きを示すフローチャートである。

【図50】本発明の符号化プロセスの他の実施例を示すフローチャートである。

【図51】図50に示したフローチャートの続きを示すフローチャートである。

【図52】本発明の復号プロセスの他の実施例を示すフローチャートである。

【図53】図52に示したフローチャートの続きを示すフローチャートである。

【図54】本発明の符号化プロセス及び復号プロセスにおける各係数のモデリングのためのプロセスの他の実施例を示すフローチャートである。

【図55】図54に示したフローチャートの続きを示すフローチャートである。

【図56】本発明において係数アライメントのために用いられる周波数帯域用乗数の例を示す図である。

【図57】1レベル分割RTS変換用のフォワードフィルタを実現するためのC言語ソースリストの一例を示す図である。

【図58】1レベル分割RTS変換用のインバースフィルタを実現するためのC言語ソースリストの一例を示す図である。

【図59】2レベル分割RTS変換用のフォワードフィルタを実現するためのC言語ソースリストの一例を示す図である。

【図60】2レベル分割RTS変換用のインバースフィルタを実現するためのC言語ソースリストの一例を示す図である。

【符号の説明】

102 可逆ウェーブレット変換ブロック

103 ビット・シグニフィカンス埋め込みブロック

104 エントロピー符号化器

105 周波数ベースコンテキストモデル

106 統合空間/周波数(JSF)コンテキストモデル

108 スイッチ

109 符号-絶対値フォーマットユニット

401, 405, 409 低域通過フィルタユニット

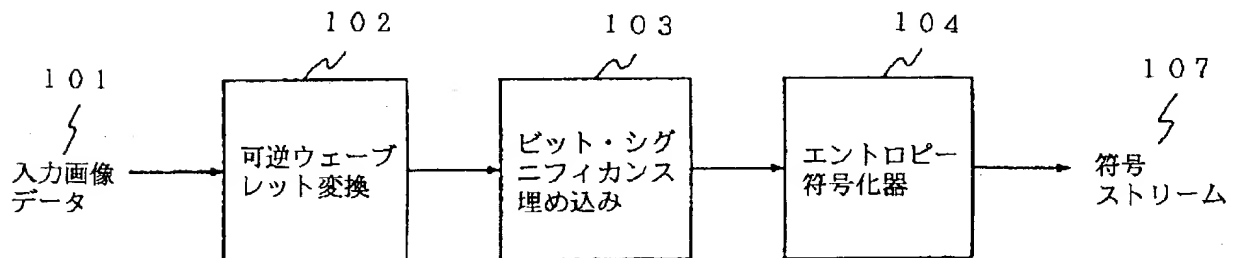
71

402, 406, 410 高域通過フィルタユニット  
 403, 404, 407 サブサンプリングユニット  
 408, 411, 412 サブサンプリングユニット  
 1000 フォワード・ウェーブレットフィルタ  
 1001, 1006 加算器  
 1002, 1005 減算器  
 1003 1ビットシフトブロック  
 1004 2ビットシフトブロック  
 1007 2ビットシフトブロック  
 1101 減算器  
 1102 2ビットシフトブロック  
 1103 1ビットシフトブロック  
 1104 減算器  
 1105 加算器  
 1106 減算器  
 1107, 1108 クリッピングブロック  
 1301 フィルタ制御ユニット  
 1302 メモリ  
 1303 フィルタ  
 1401 メモリインターフェイスユニット  
 1700 コンテキストモデル  
 1701 絶対値メモリ

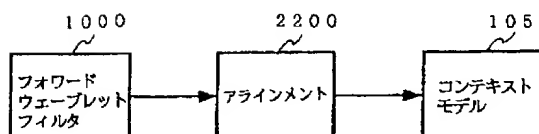
72

1702 シグニフィカンスユニット  
 1703 ツリーメモリ  
 1801 インバータ  
 1802 マルチプレクサ  
 1803 プライオリティエンコーダ  
 1804 カウンタ  
 1900~1916 カウンタ  
 1920~1936 メモリ  
 2001 インデックスカウンタ  
 2002 一致論理  
 2003 インデックスレベル論理  
 2004 シグニフィカンスカウンタ  
 2005~2007 フリップフロップ (フラグ0~2)  
 2008 デシジョンジェネレータ  
 2101 メモリ  
 2102 カウンタ  
 2103 マルチプレクサ  
 2110~2112 レジスタ  
 2200 アラインメントユニット  
 2301 "1.5" プライオリティエンコーダ  
 2302 マルチプレクサ

【図1】

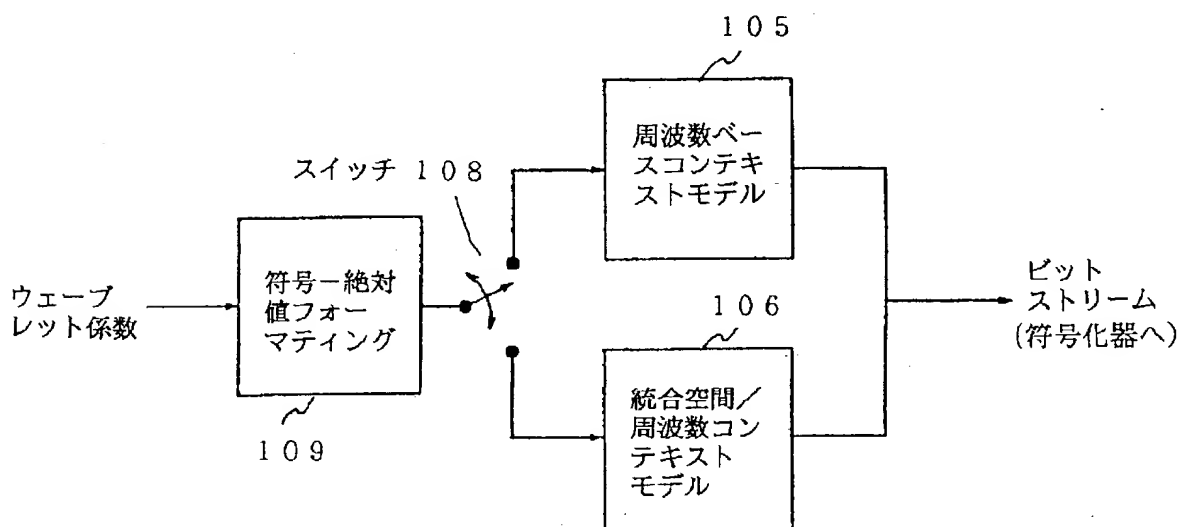


【図41】

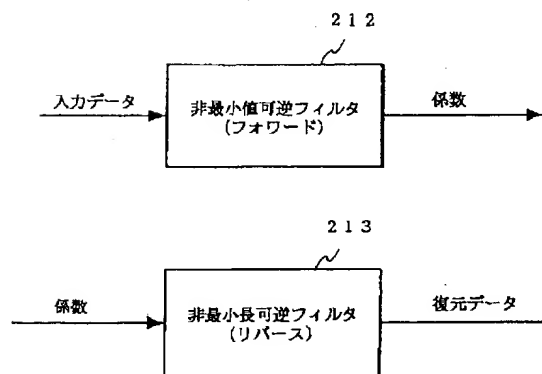




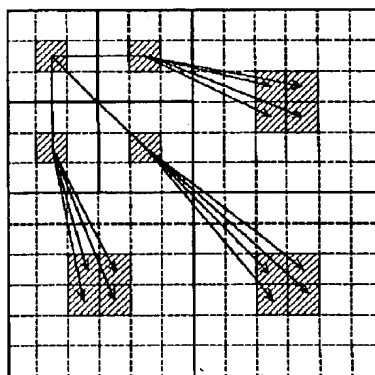
【図2】



【図4】

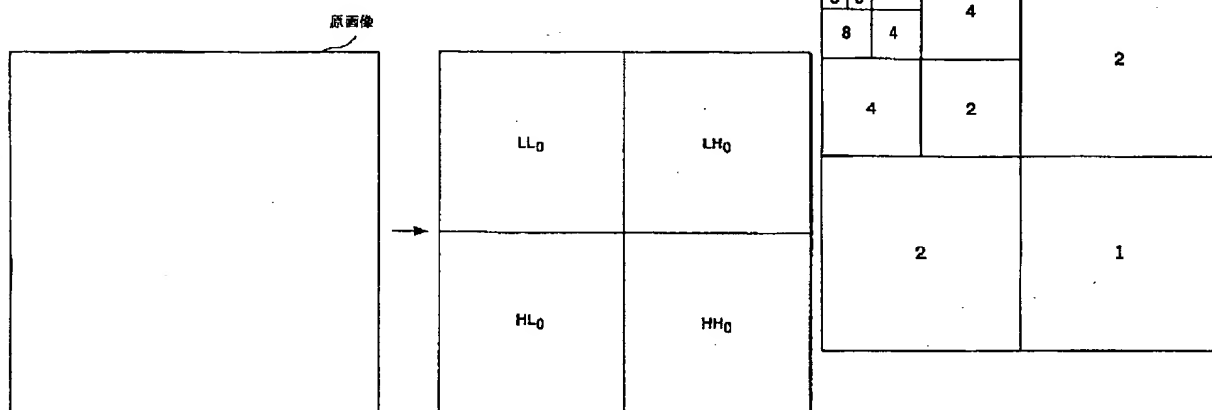


【図14】

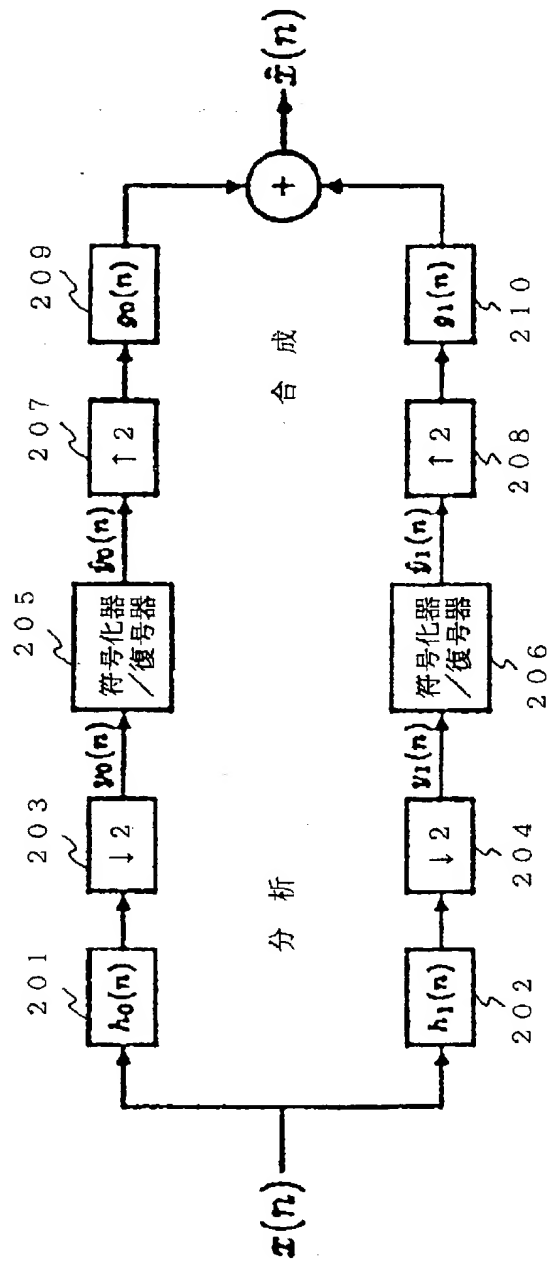


【図56】

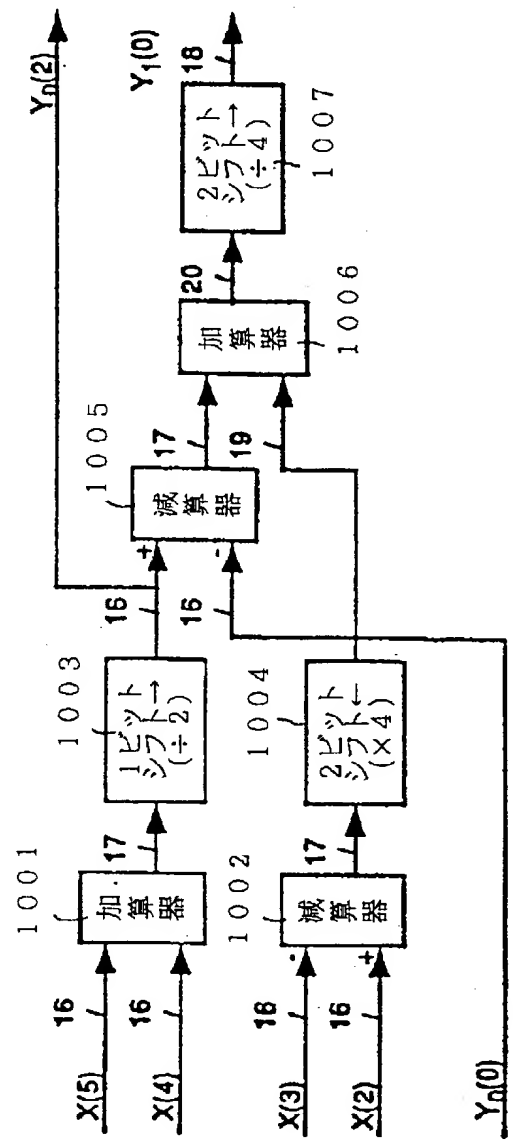
【図5】



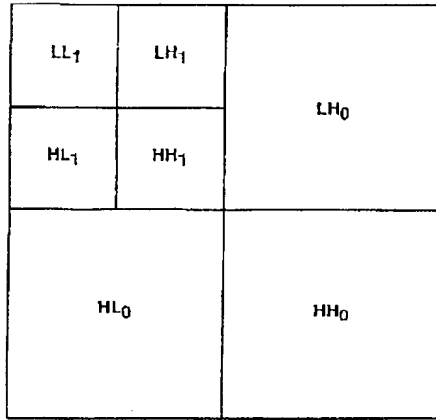
【図3】



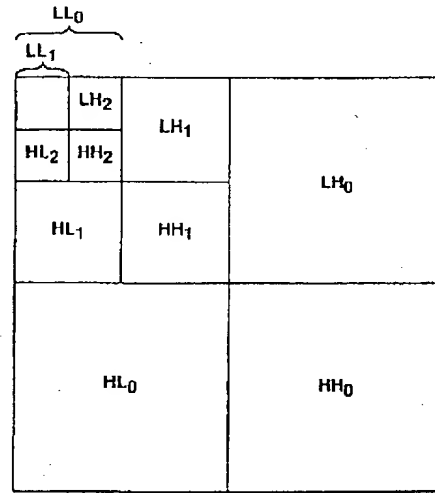
【図29】



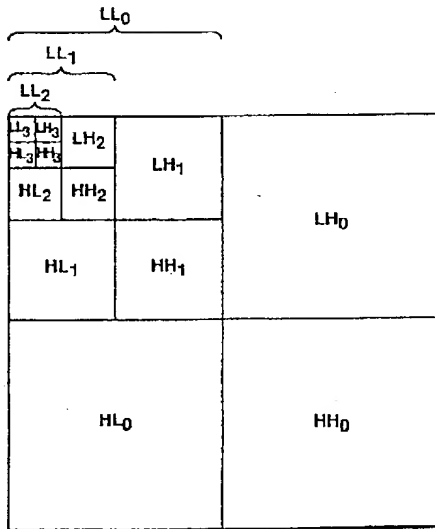
【図6】



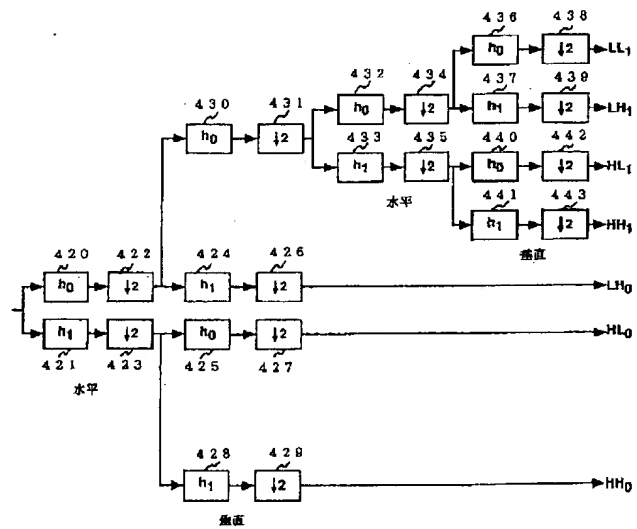
【図7】



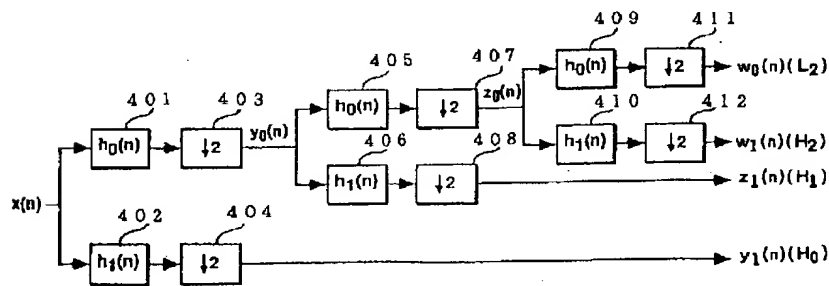
【図8】



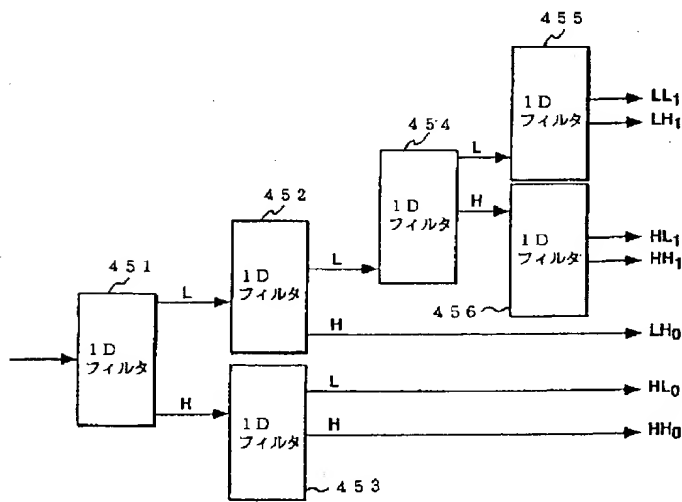
【図10】



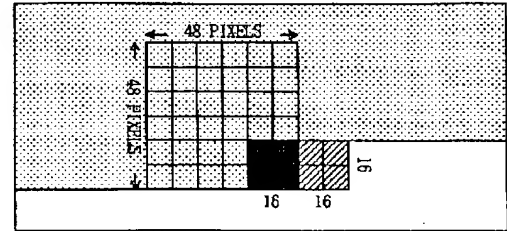
【図9】



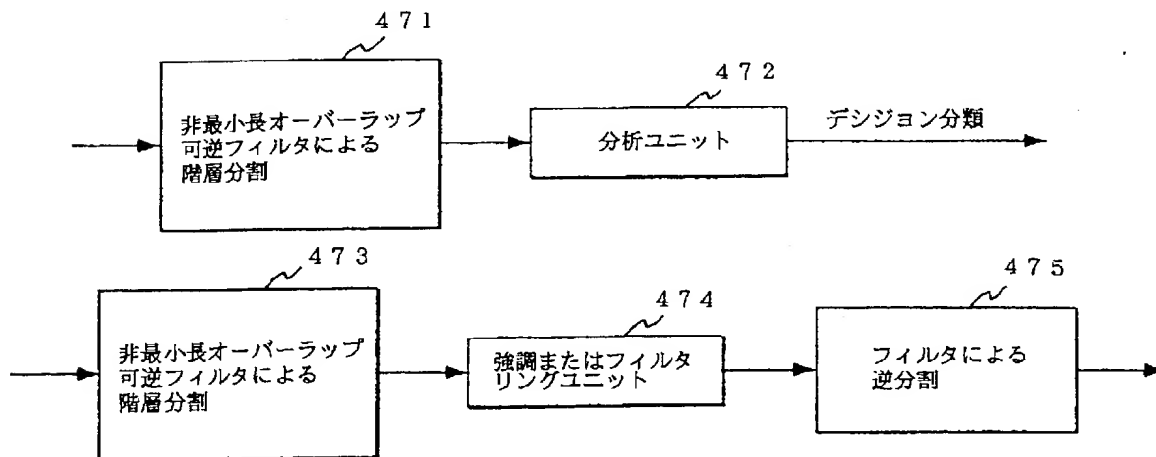
【図 11】



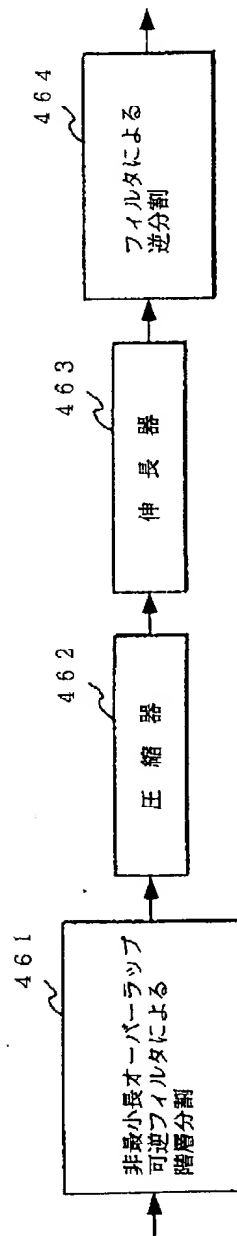
【図 31】



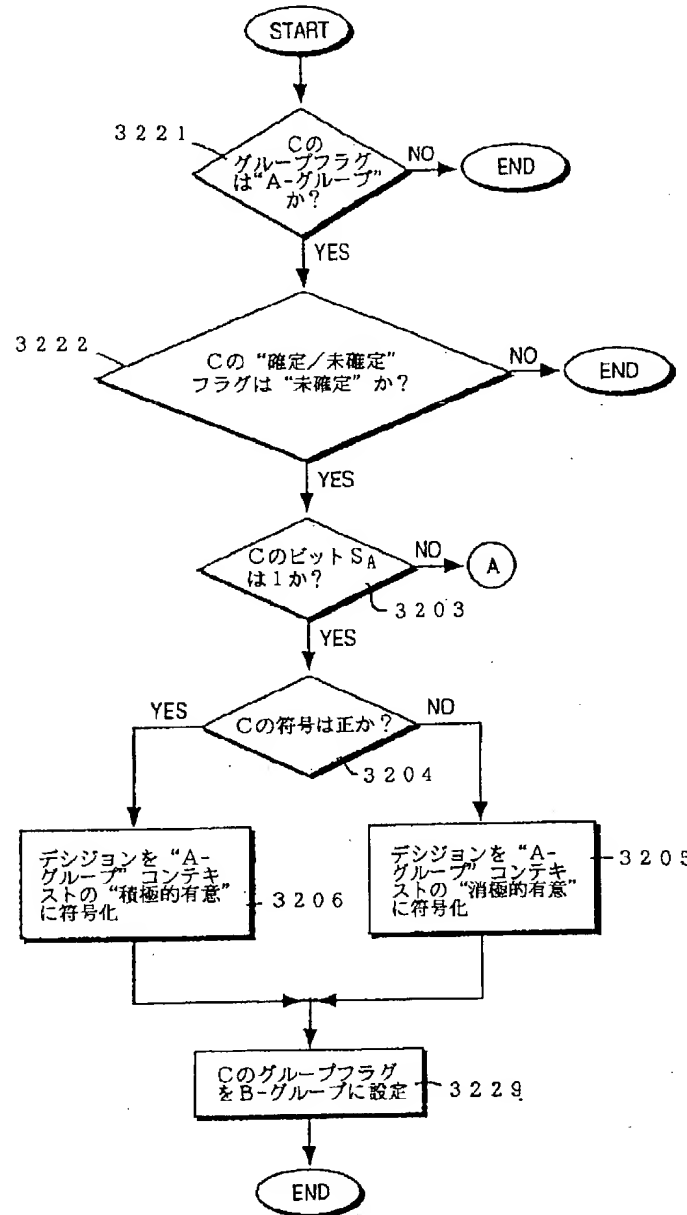
【図 13】



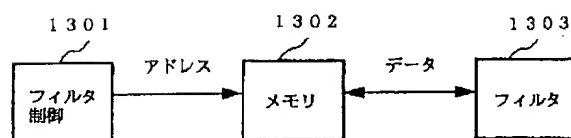
【図12】



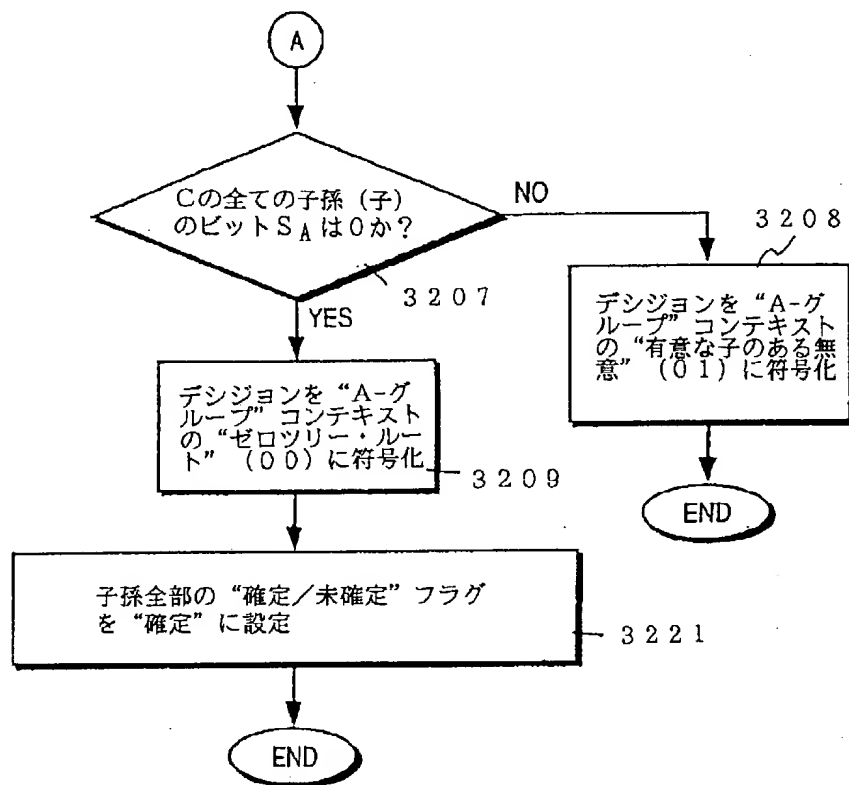
【図15】



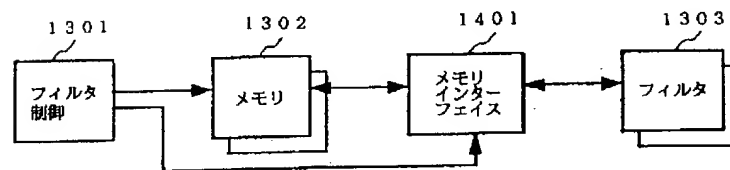
【図32】



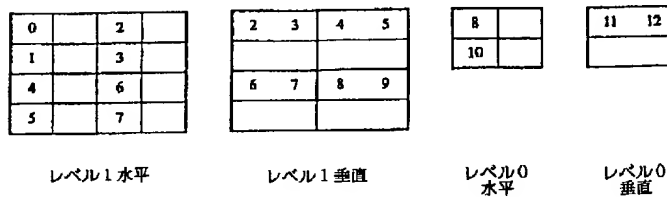
【図16】



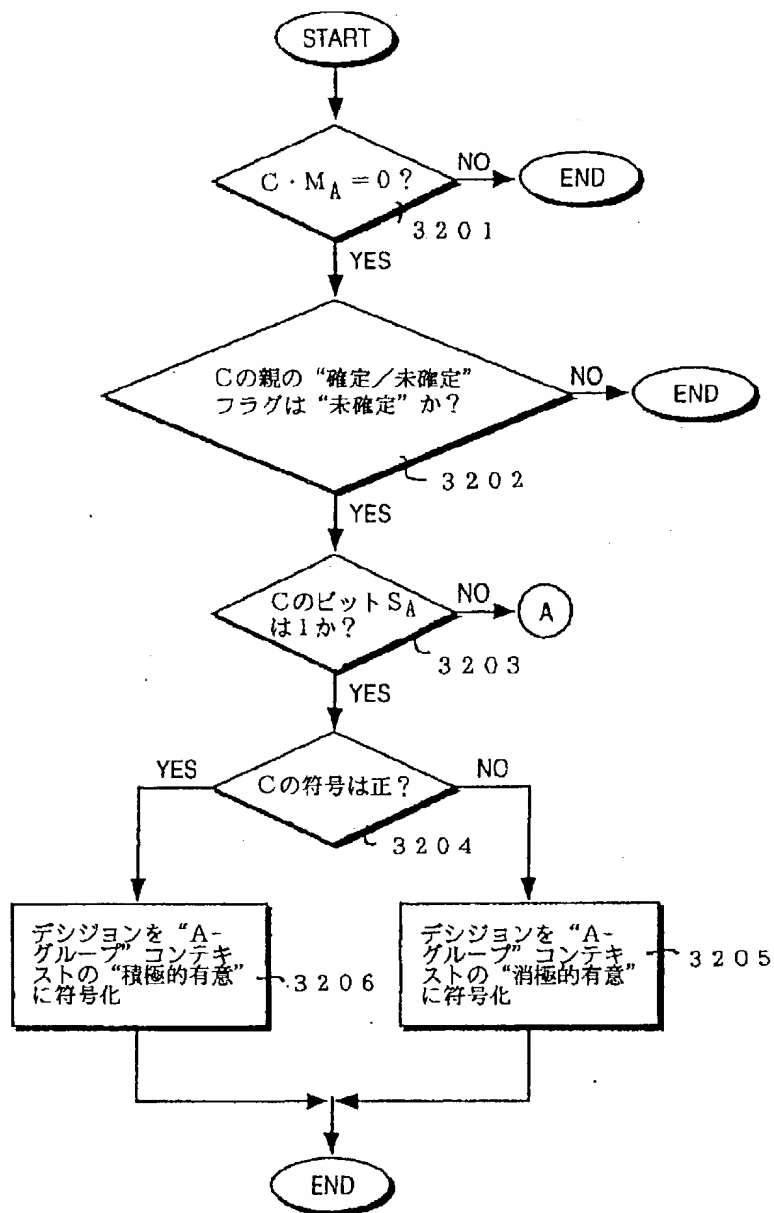
【図33】



【図35】

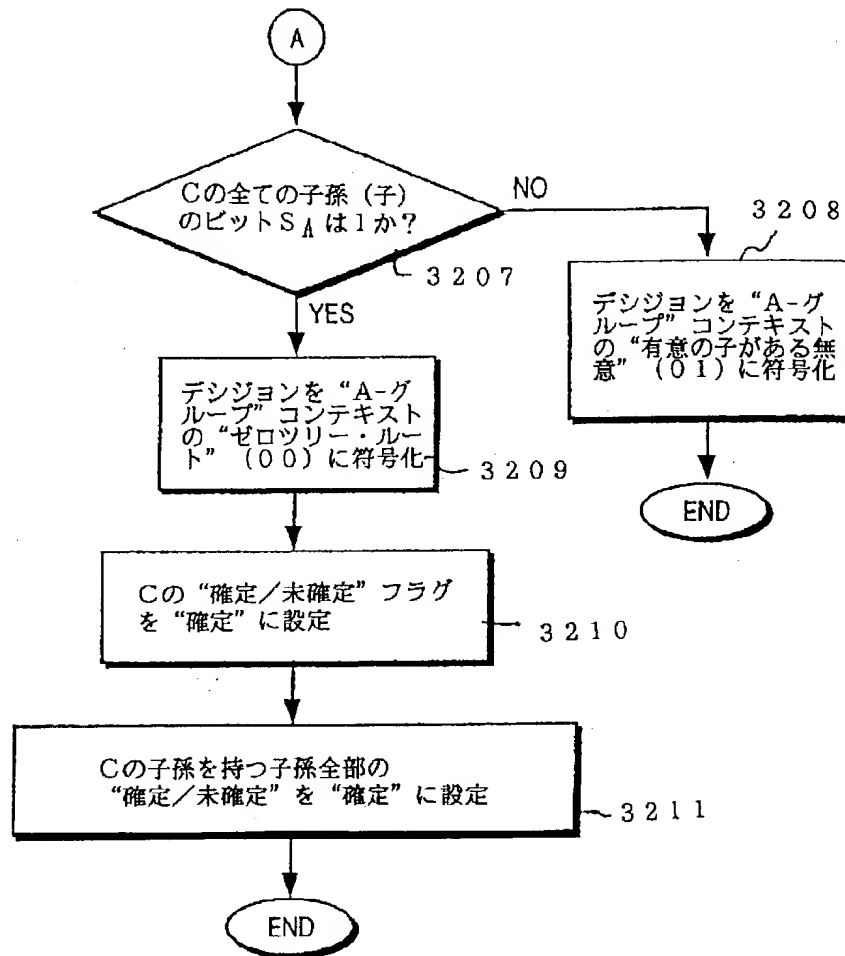


【図 17】





【図18】



【図34】

(2バンク)

0	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0

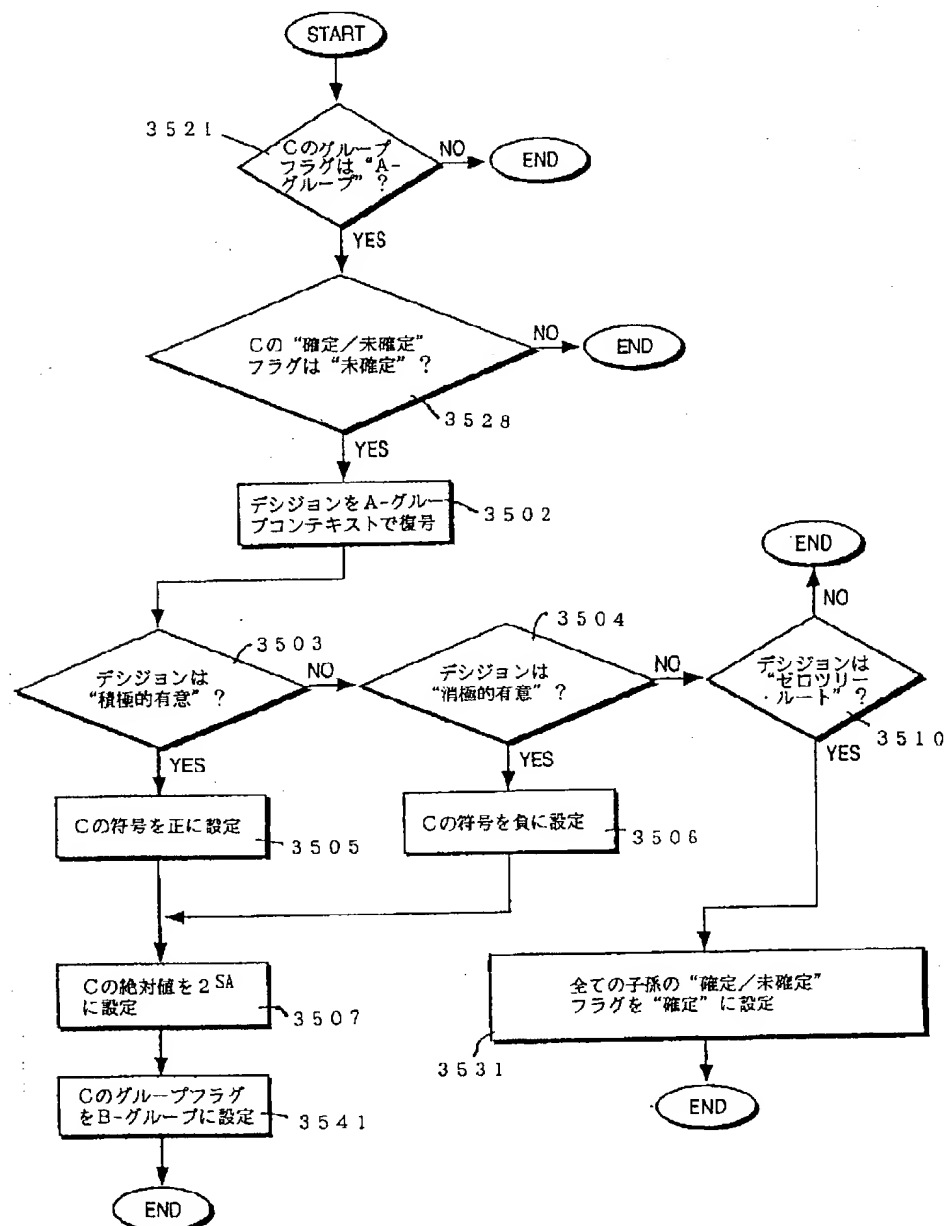
...

(4バンク)

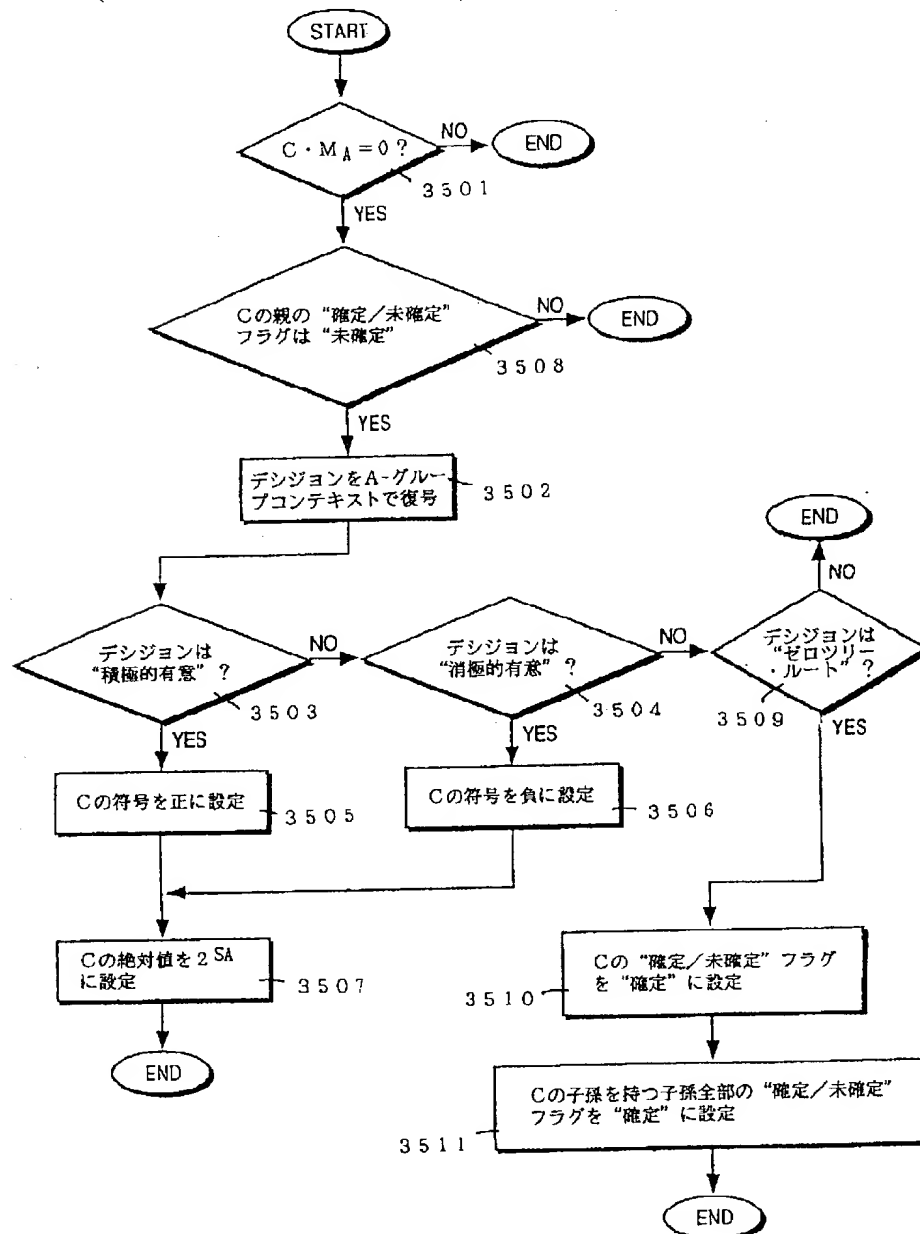
0	1	2	3	0	1
1	2	3	0	1	2
2	3	0	1	1	3
3	0	1	2	3	0
0	1	2	3	0	1
1	2	3	0	1	2

...

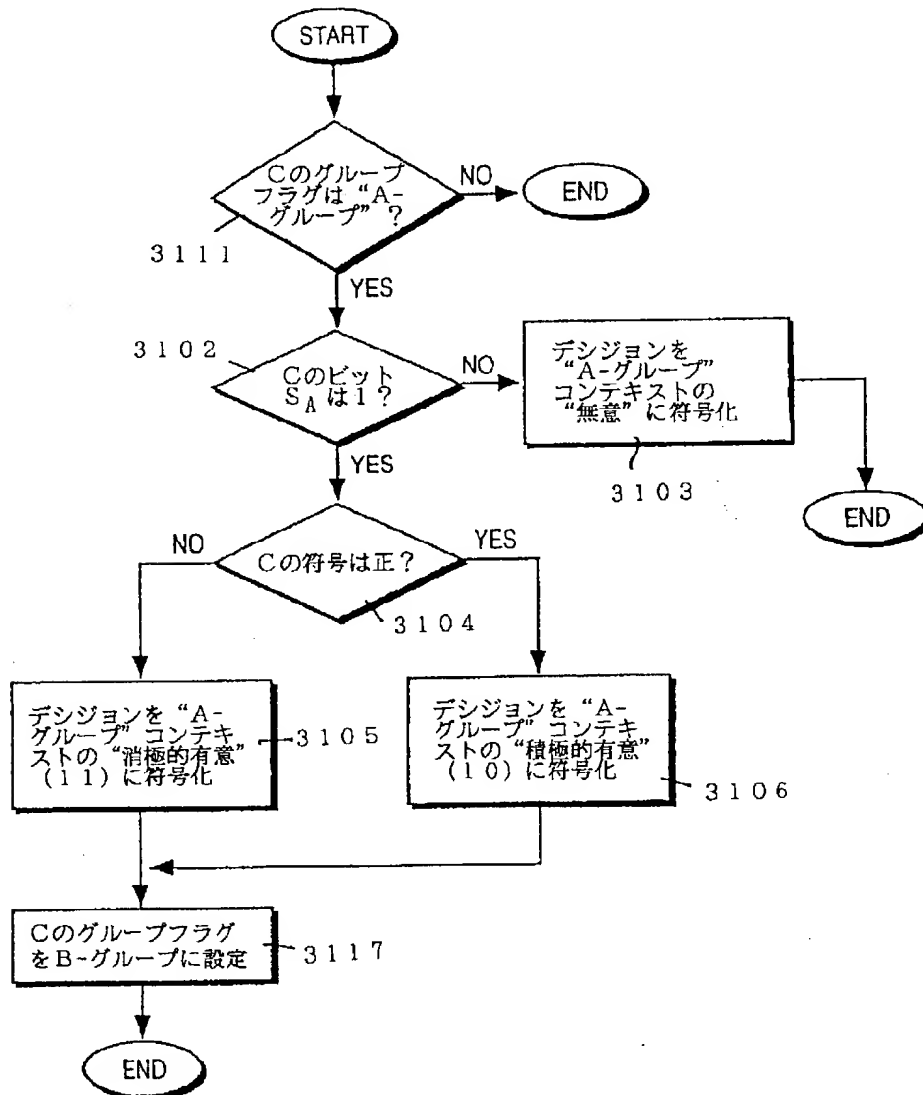
【図19】



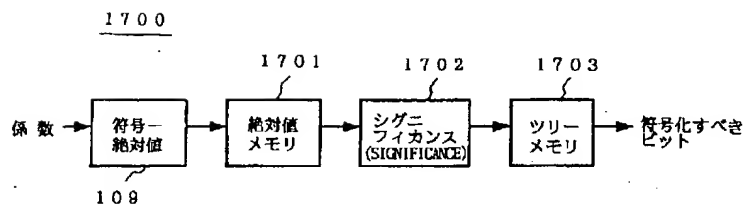
【図20】



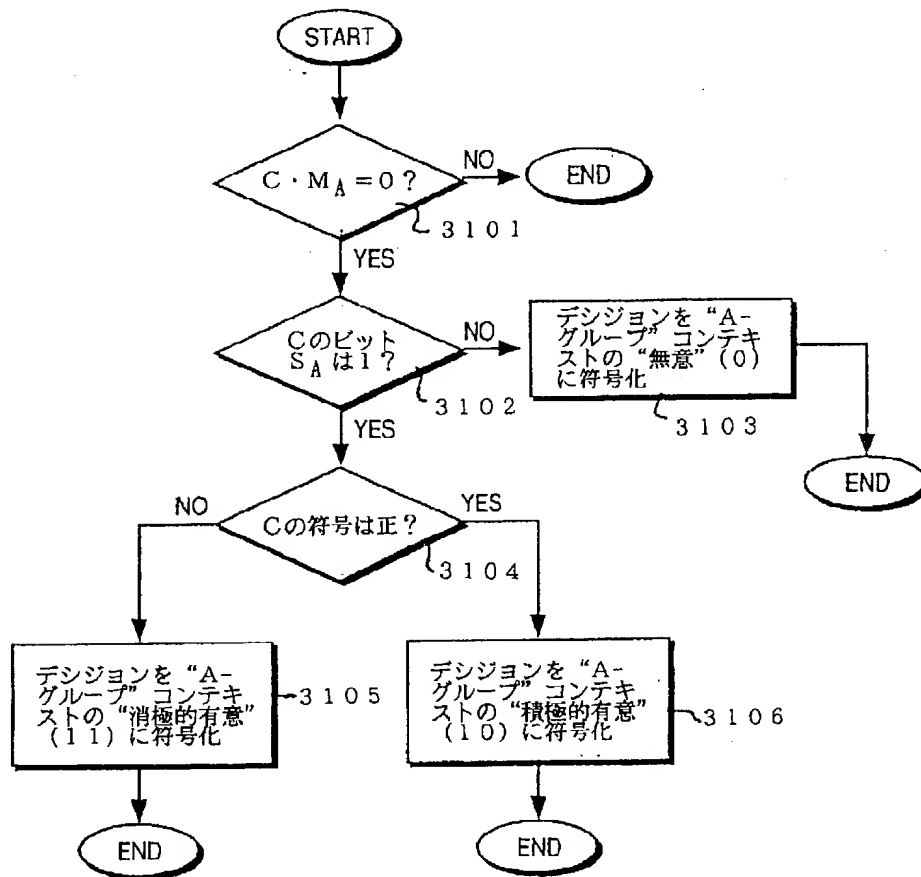
【図21】



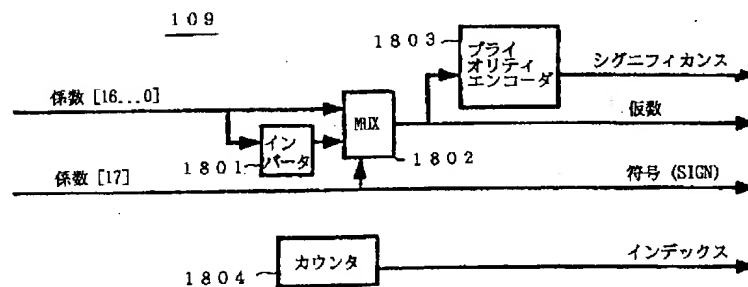
【図36】



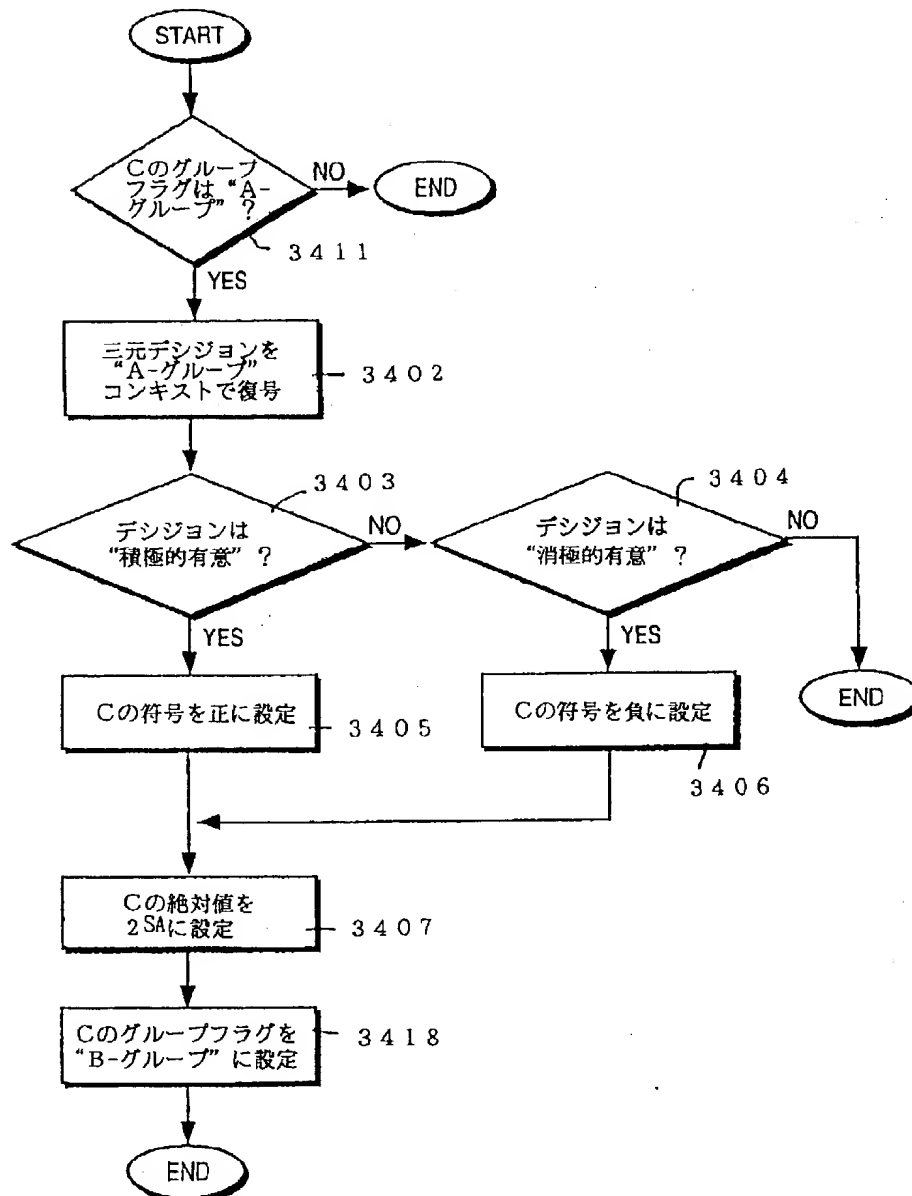
【図 22】



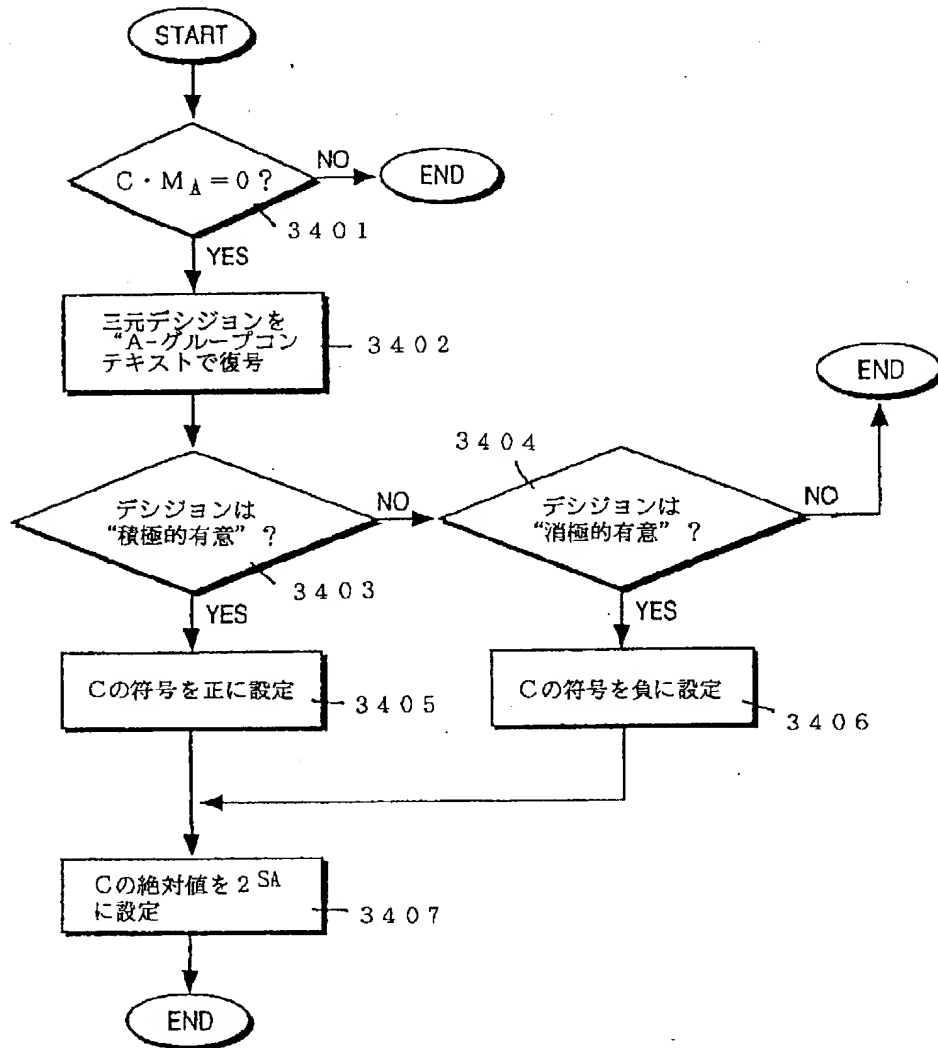
【図 37】



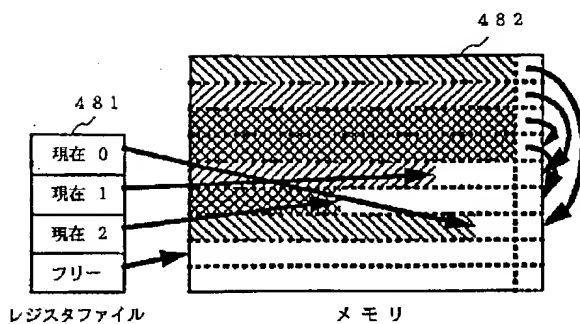
【図23】



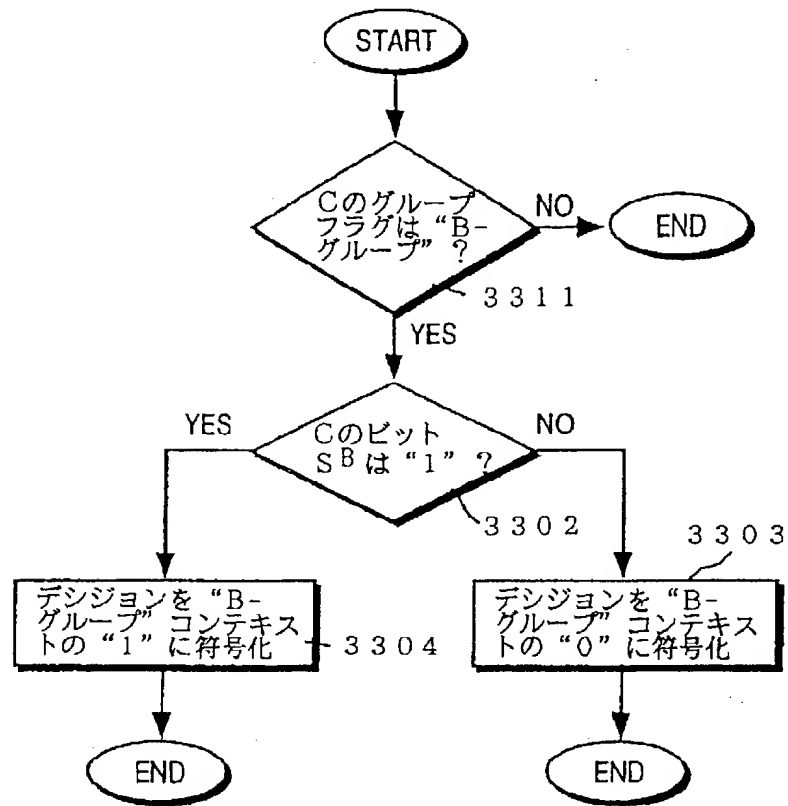
【図 24】



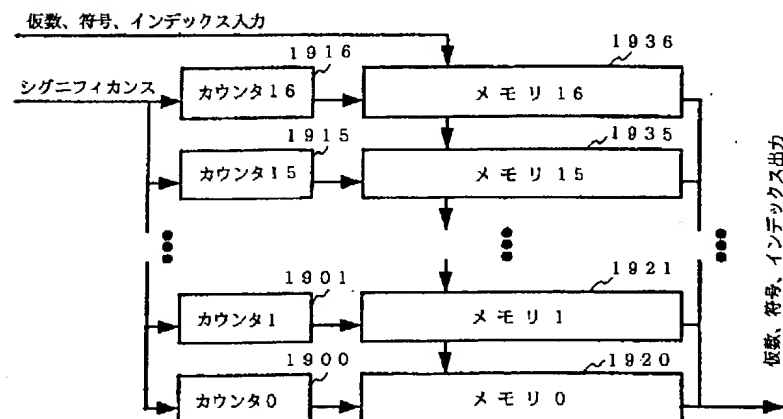
【図 43】



【図25】

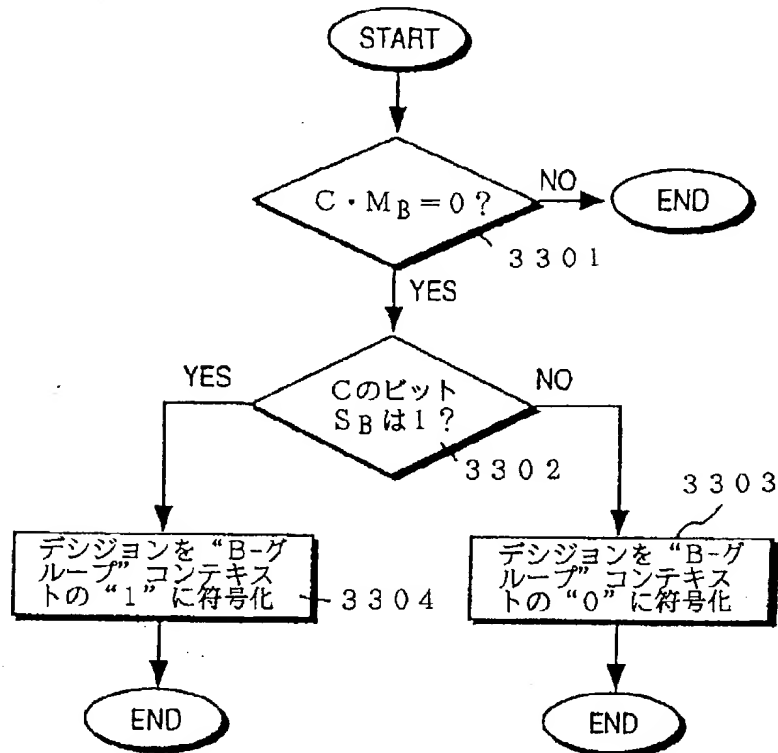


【図38】

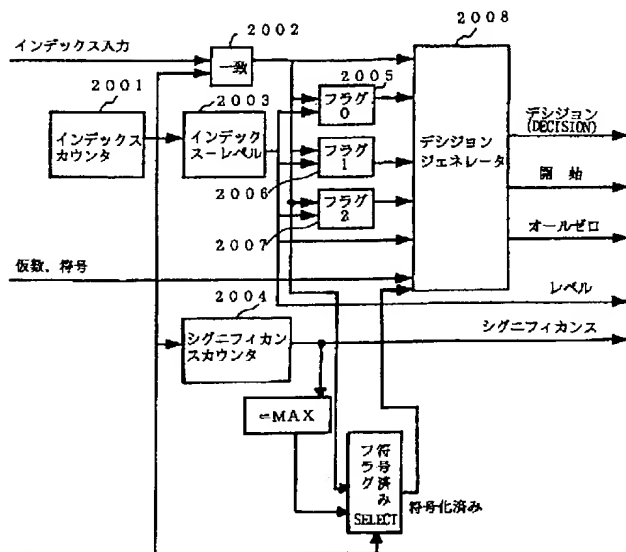




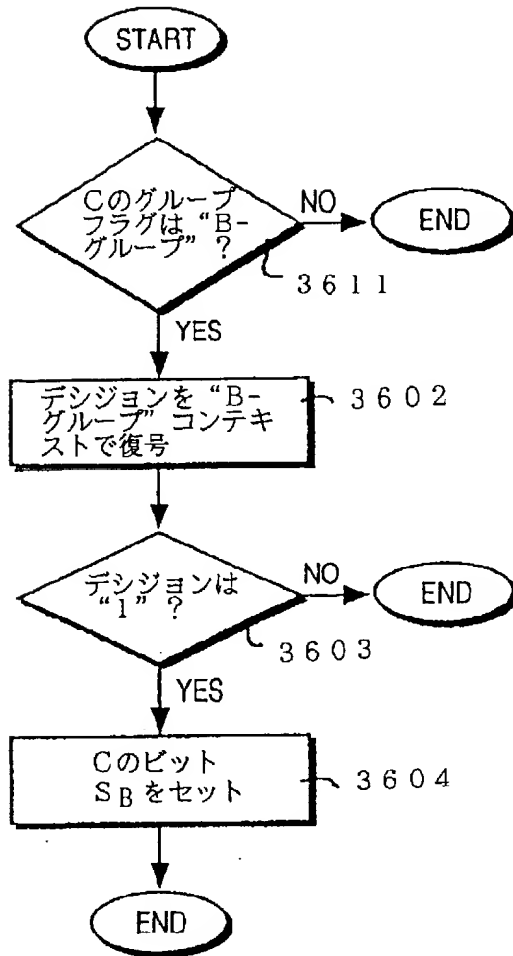
【図26】



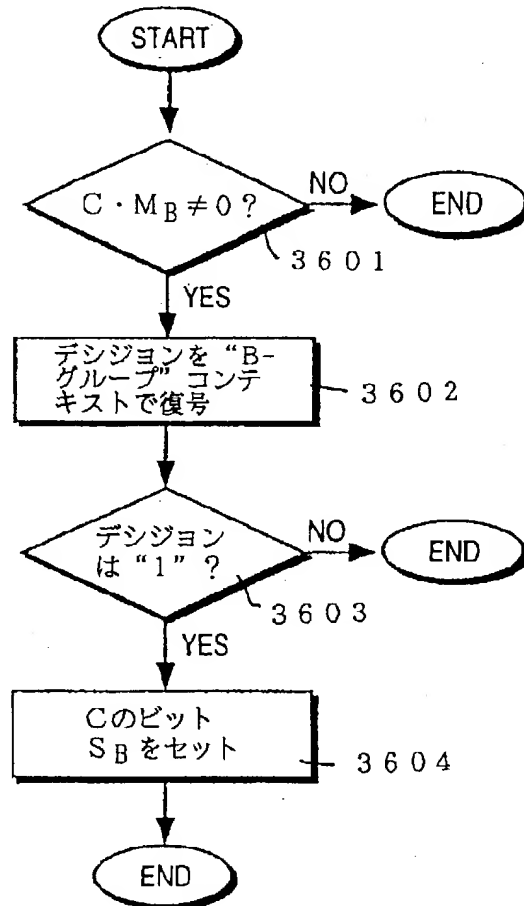
【図39】



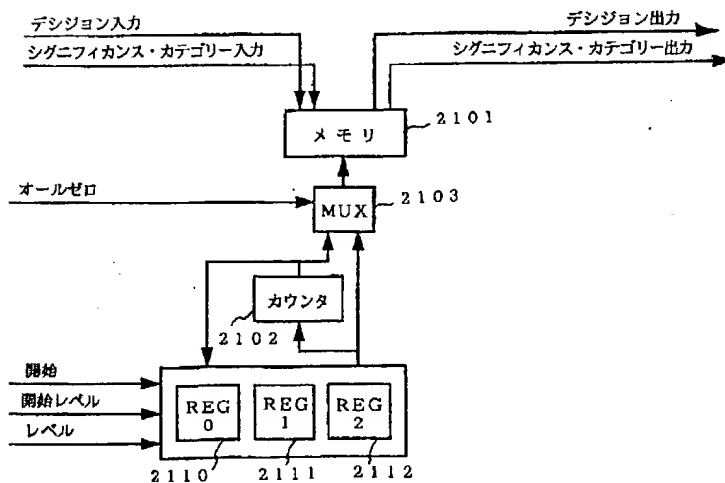
【図27】



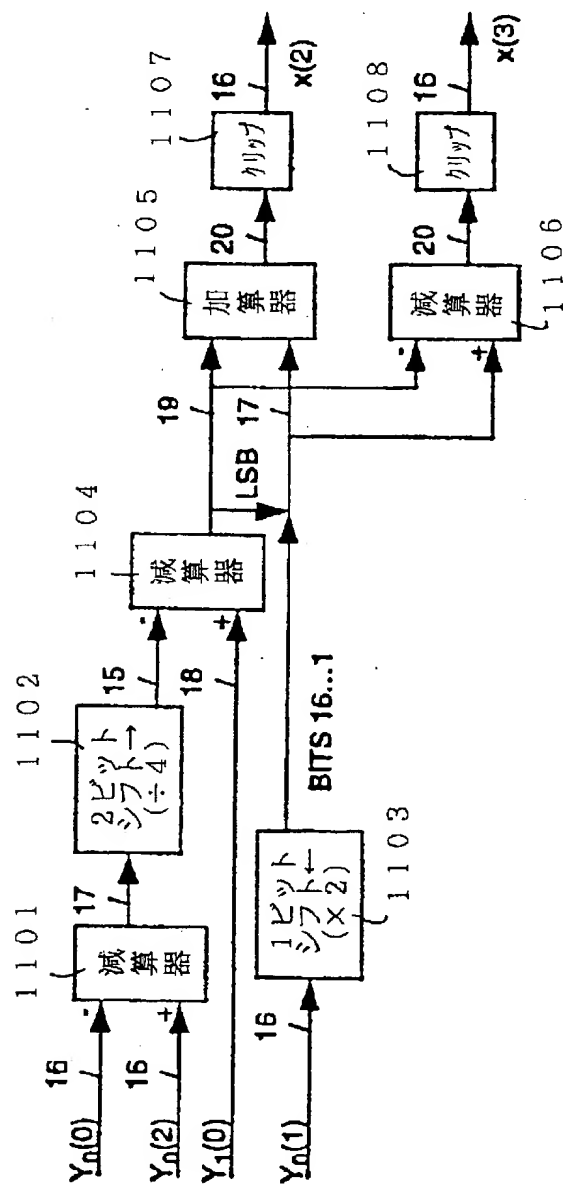
【図28】



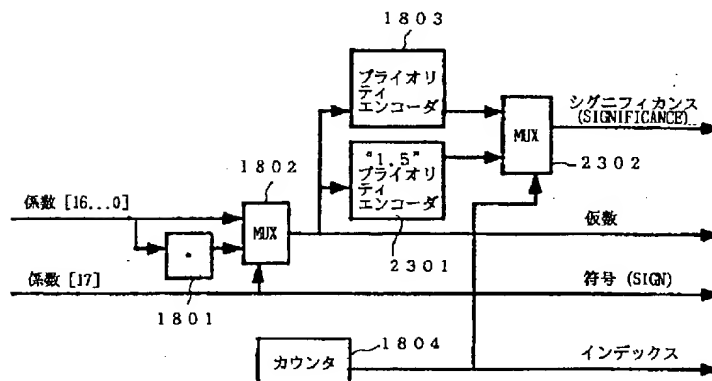
【図40】



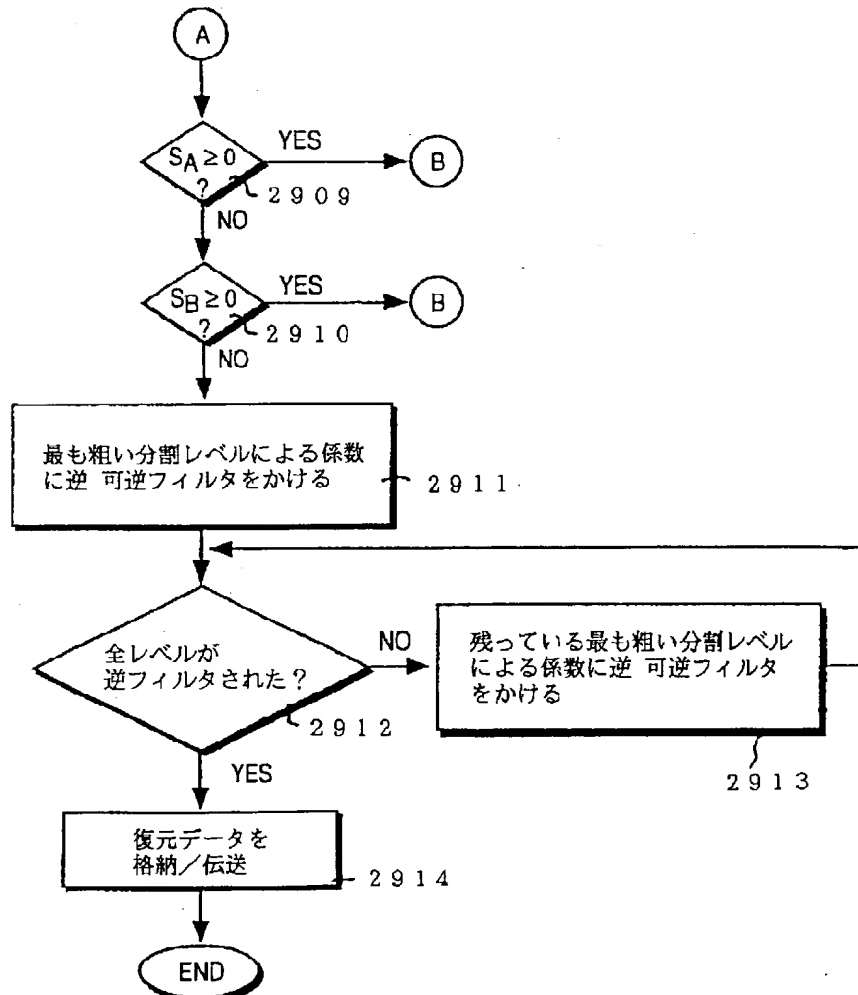
【図30】



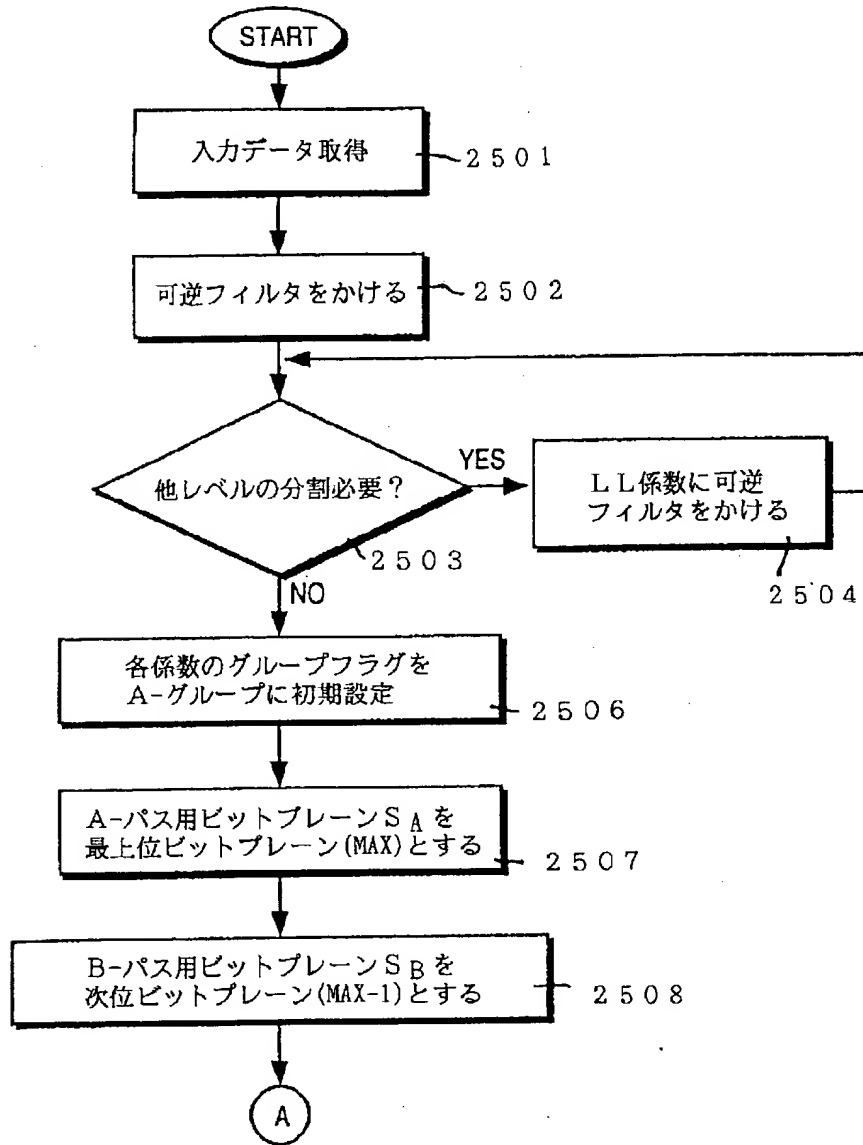
【図 42】



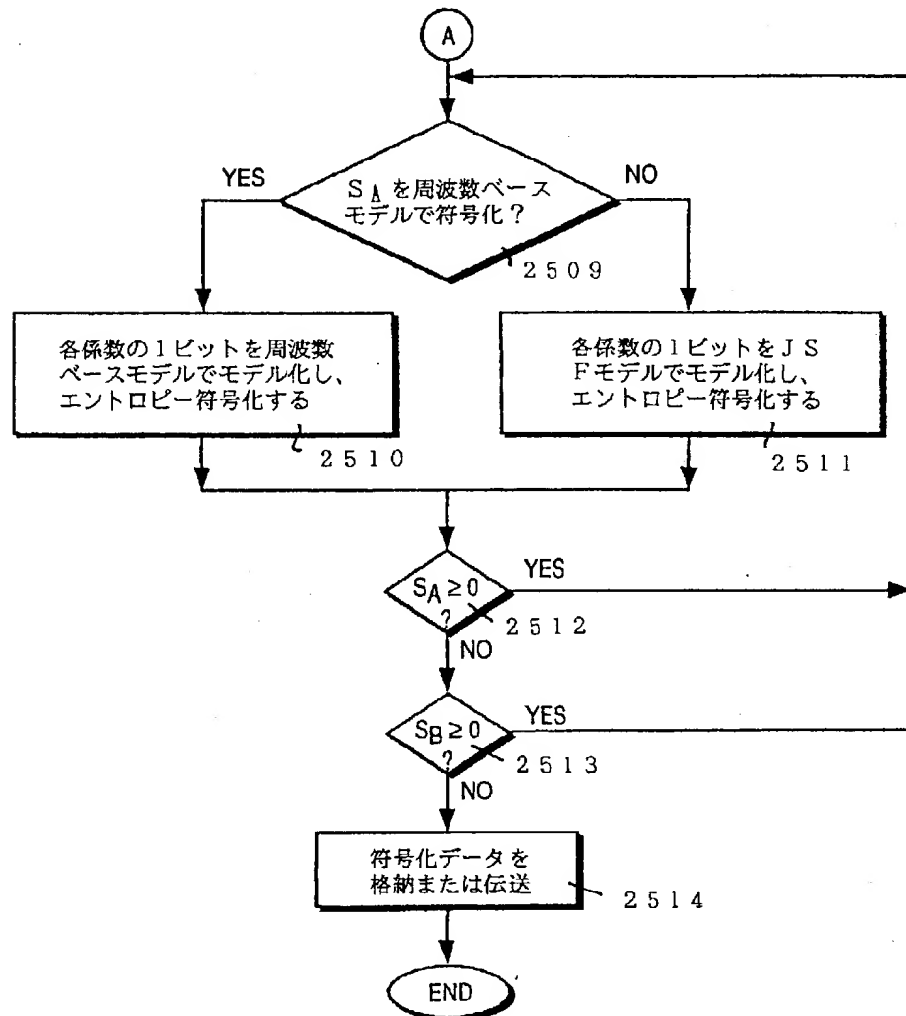
【図 53】



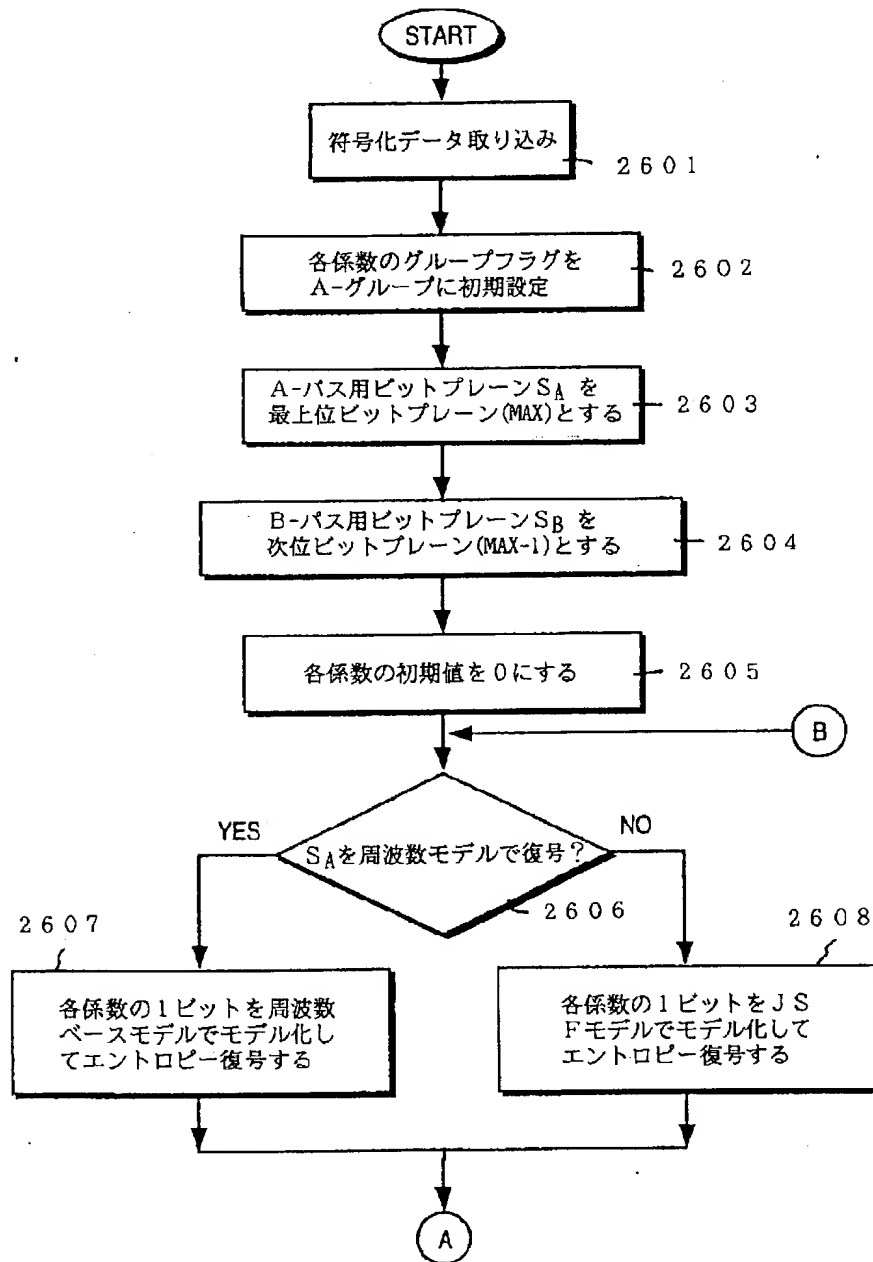
【図 44】



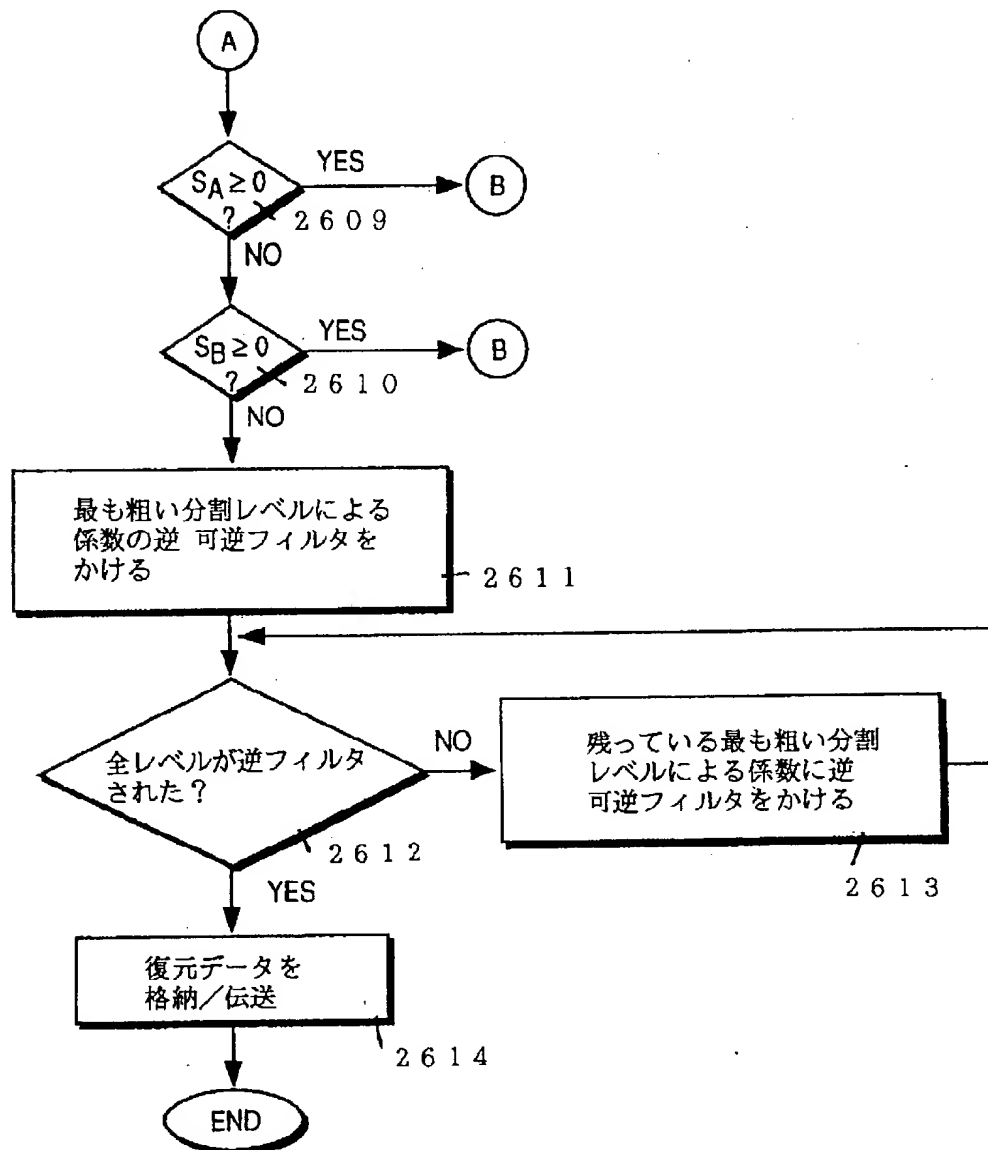
【図 45】



【図46】

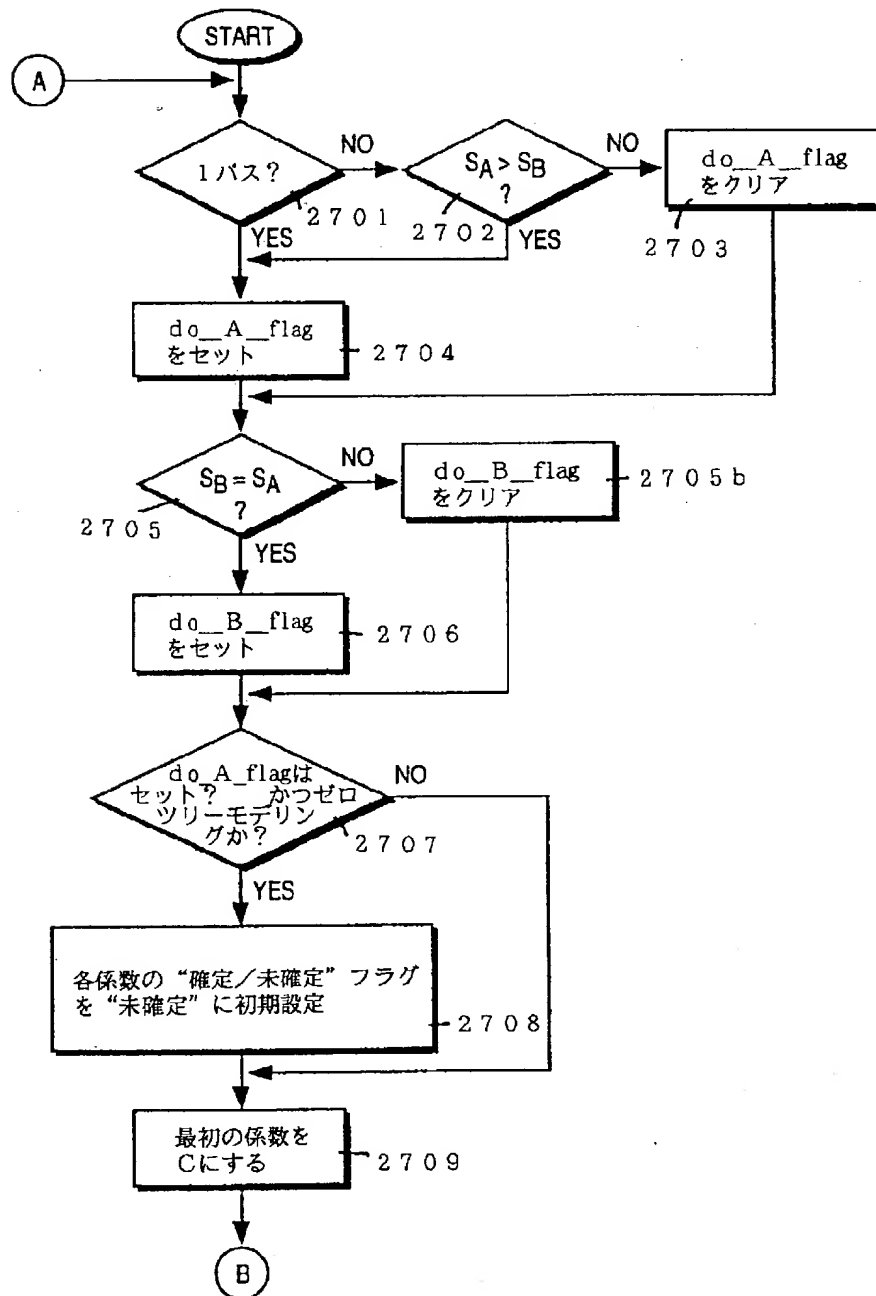


【図 47】

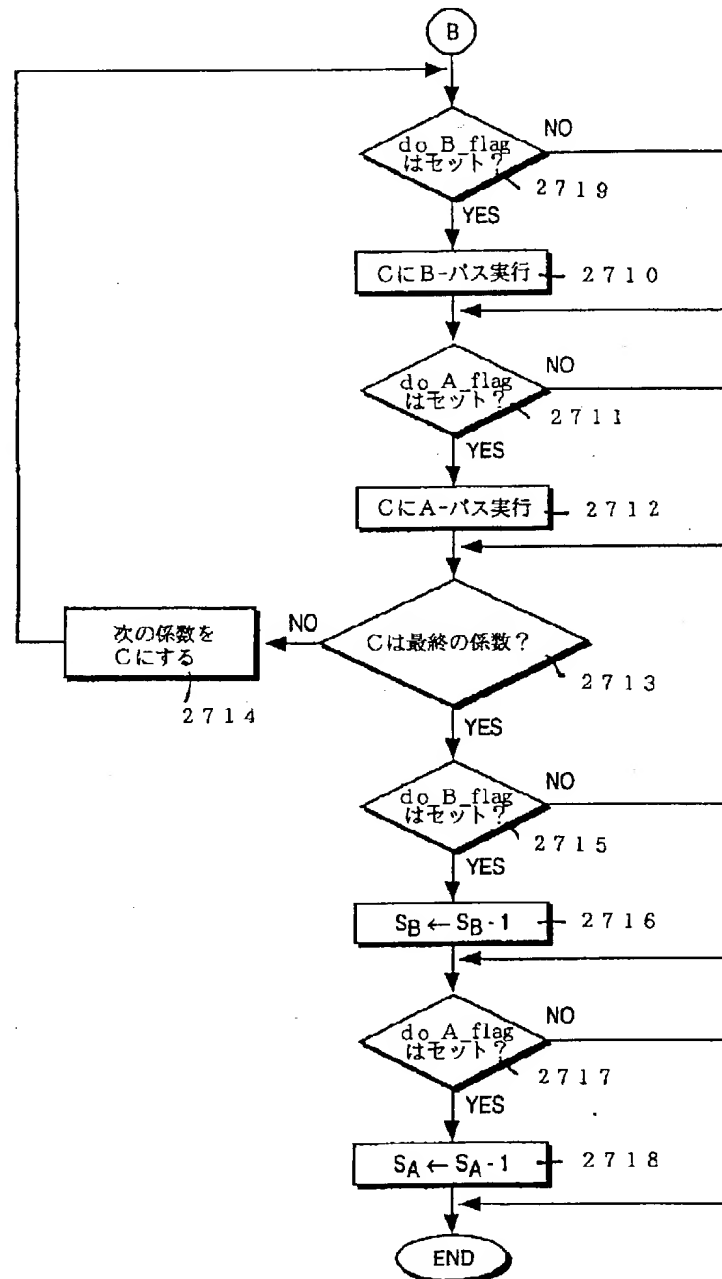




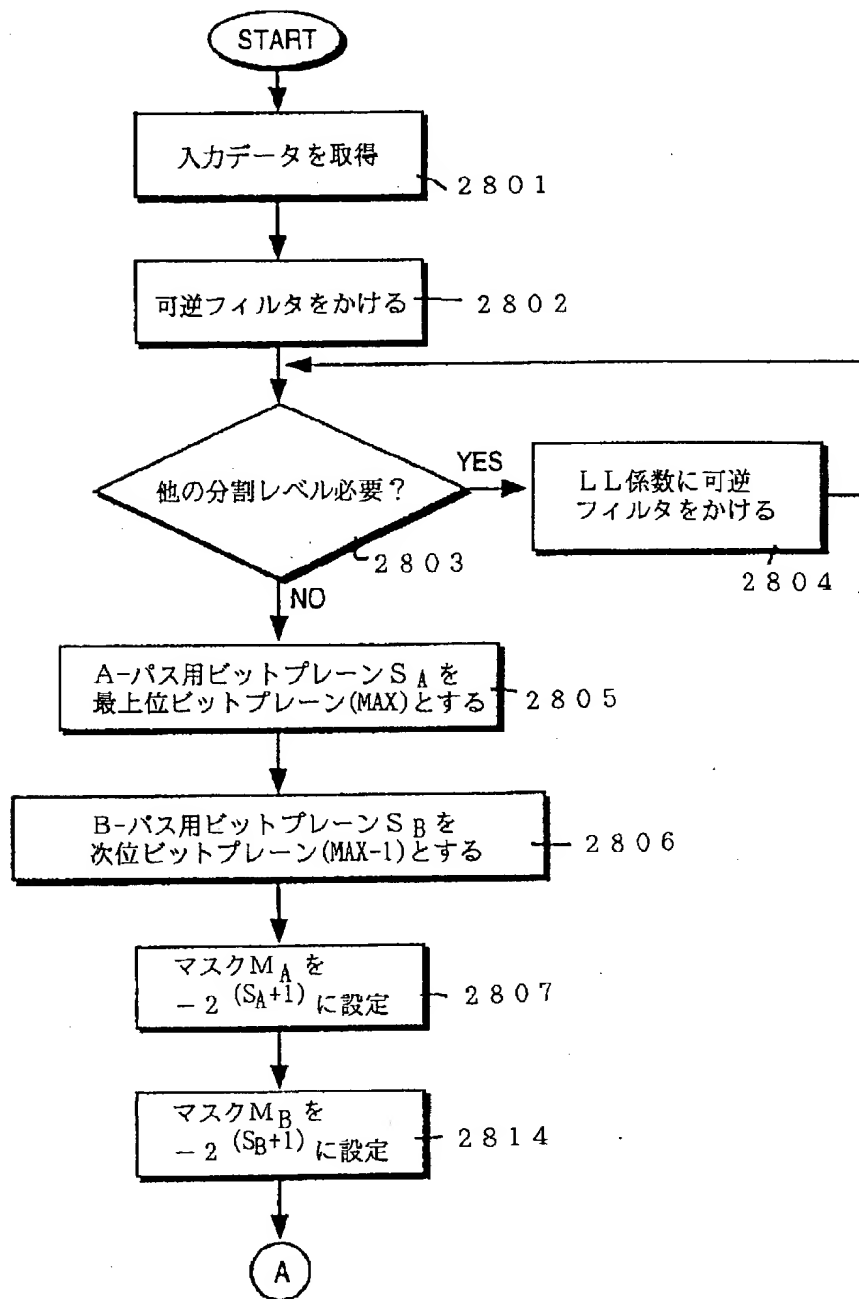
【図48】



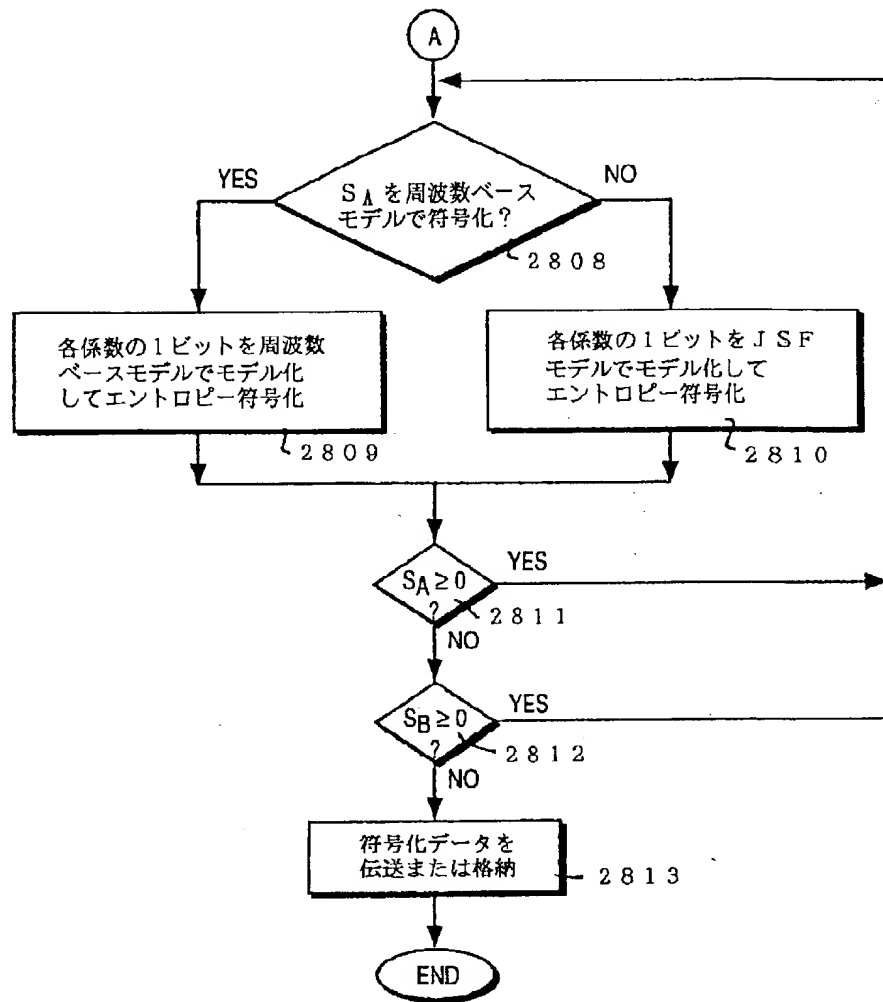
【図49】



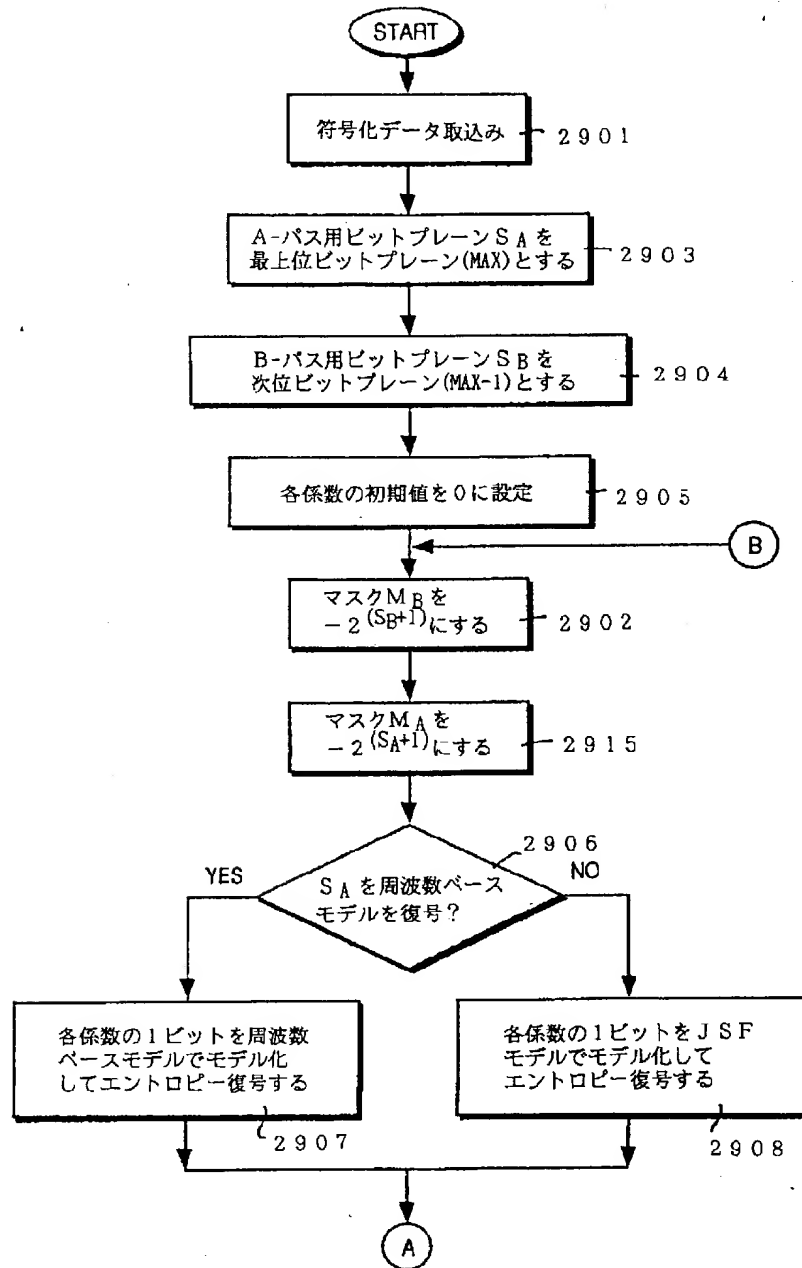
【図50】



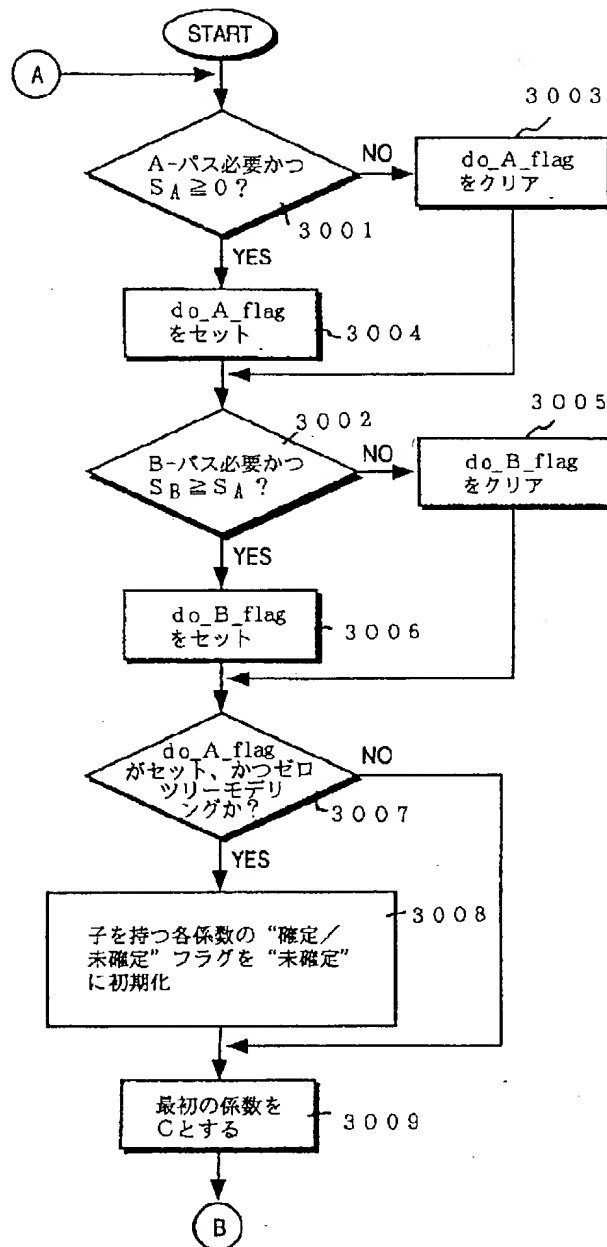
【図 51】



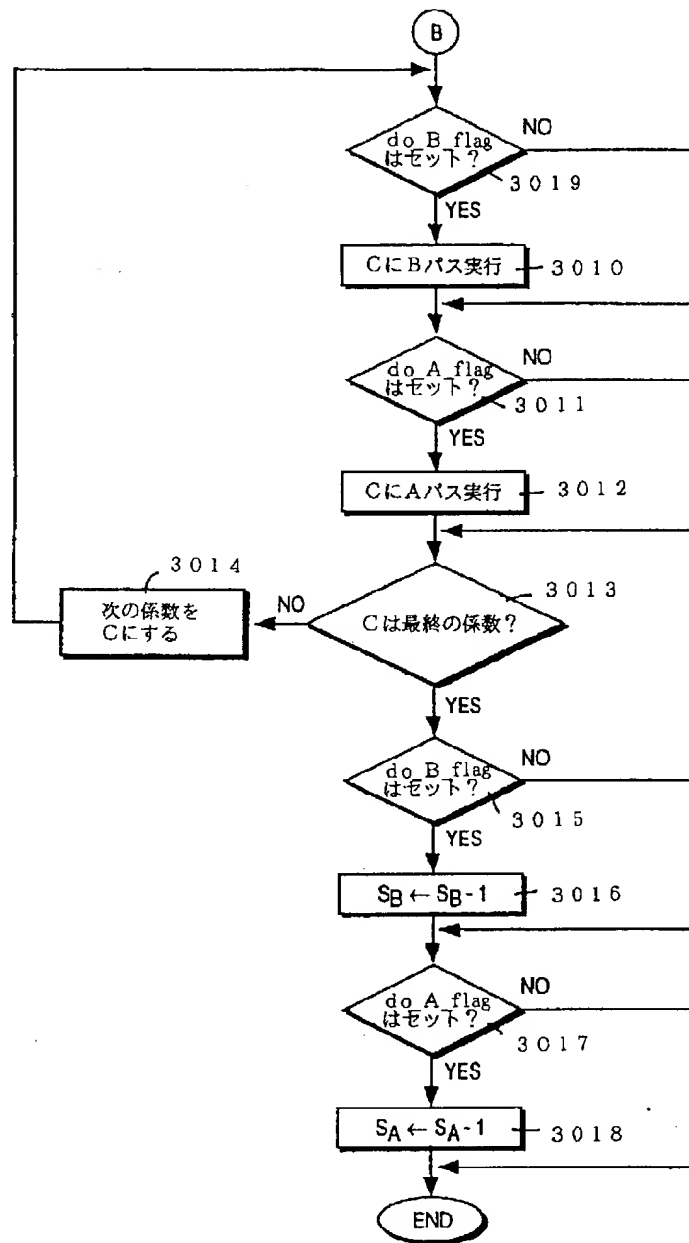
【図52】



【図 54】



【図55】



【図57】

## RTS変換 (フォワード) - 1レベル分割

```

void RTS(T,w,m,n)
int w,m,n;
int *T;
{
    int i,j,k;
    int *X, *Y;

    for (i=0;i<n;i++)
    {
        for (j=0;j<m;j+=2)
        {
            X[j/2] = (T[i*w+j]+T[i*w+j+1])>>1;
            Y[j/2] = T[i*w+j]-T[i*w+j+1];
        }
        for (j=0;j<m/2;j++)
        {
            T[i*m+j]=X[j];
            T[i*m+m/2+j]= (-X[j]+(Y[(j+1)%m/2]<<2)+X[(j+2)%m/2])>>2;
        }
    }
}

```

【図58】

## RTS変換 (インバース) - 1レベル分割

```

void InverseRTS(T,w,m,n)
int w,m,n;
int *T;
{
    int i,j;
    int *B, *D;

    for (i=0;i<n;i++)
    {
        for (j=0;j<m/2;j++)
        {
            B[j] = (-T[i*m+j]+T[i*m+((j+2)%m/2)])&0X3;
            D[(j+1)%m/2] = ((T[i*m+j]-T[i*m+((j+2)%m/2)])+
                            (T[i*m+m/2+j]<<2)+B[j])>>2;
        }
        for (j=0;j<m/2;j++)
        {
            T[i*w+2*j] = ((T[i*m+j]<<1)+D[j+1])>>1;
            T[i*w+2*j+1] = T[i*w+2*j] - D[j];
        }
    }
}

```



【図59】

## RTS変換 (フオワード) - 2レベル分割

```

void RTS(T,w,m,n)
int w,m,n;
int *T;
{
    int i,j,k;
    int *temp;
    int *X, *Y;

    for (i=0; i<n; i+=2)
    {
        for (j=0; j<m; j+=2)
        {
            X[j/2] = (T[i*w+j]+T[i*w+j+1])>>1;
            Y[j/2] = T[i*w+j]-T[i*w+j+1];
        }
        for (j=0; j<m/2; j++)
        {
            temp[i*m+j]=X[j];
            temp[i*m+m/2+j]= (-X[j]+(Y[(j+1)%(m/2)]<<2)+X[(j+2)%(m/2)])>>2;
        }
    }

    for (j=0; j<m; j++)
    {
        for (i=0; i<n; i+=2)
        {
            X[i/2] = (temp[i*m+j]+temp[(i+1)*m+j])>>1;
            Y[i/2] = temp[i*m+j]-temp[(i+1)*m+j];
        }
        for (i=0; i<n/2; i++)
        {
            T[i*w+j]=X[i];
            T[(i+n/2)*w+j]= (-X[i]+(Y[(i+1)%(n/2)]<<2)+X[(i+2)%(n/2)])>>2;
        }
    }
}

```

【図60】

## RTS変換 (インバース) - 2レベル分割

```

void InverseRTS(T,w,m,n)
int w,m,n;
int *T;
{
    int i,j;
    int *B, *D;
    int *temp;

    for (j=0;j<m;j++)
    {
        for (i=0;i<n/2;i++)
        {
            B[i] = (-T[i*w+j]+T[((i+2)&(n/2))*w+j])&0X3;
            D[(i+1)&(n/2)] = ((T[i*w+j]-T[((i+2)&(n/2))*w+j]+(T[(n/2+i)*w+j]<<2)+B[i])>>2);
        }
        for (i=0;i<n/2;i++)
        {
            temp[2*i*m+j] = ((T[i*w+j]<<1)+D[i+1])>>1;
            temp[(2*i+1)*m+j] = temp[2*i*m+j] - D[i];
        }
    }

    for (i=0;i<n;i++)
    {
        for (j=0;j<m/2;j++)
        {
            B[j] = (-temp[i*m+j]+temp[i*m+((j+2)&(m/2))])&0X3;
            D[(j+1)&(m/2)] = ((temp[i*m+j]-temp[i*m+((j+2)&(m/2))]+
                (temp[i*m+m/2+j]<<2)+B[j])>>2);
        }
        for (j=0;j<m/2;j++)
        {
            T[i*w+2*j] = ((temp[i*m+j]<<1)+D[j+1])>>1;
            T[i*w+2*j+1] = T[i*w+2*j] - D[j];
        }
    }
}

```

---

フロントページの続き

(72)発明者 エドワード エル シュワル  
 アメリカ合衆国 カリフォルニア州  
 94025 メンローパーク サンド ヒル  
 ロード 2882 リコーコーポレーション内

(72)発明者 マーティン ピー ボーリック  
 アメリカ合衆国 カリフォルニア州  
 94025 メンローパーク サンド ヒル  
 ロード 2882 リコーコーポレーション内